SoftCast: Exposing a Waveform Interface to the Wireless Channel for Scalable Video Broadcast

by

Szymon Kazimierz Jakubczak

M.Sc. Massachusetts Institute of Technology (2007)
mgr inz., Warsaw University of Technology (2005)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2011

Author .....
Department of Electrical Engineering and Computer Science
May 20, 2011

Certified by... .....
Dina Katabi
Associate Professor
Thesis Supervisor

Accepted by ....
Leslie A. Kolodziejski
Chairman, Department Committee on Graduate Students

SoftCast: Exposing a Waveform Interface to the Wireless Channel for Scalable Video Broadcast

by

Szymon Kazimierz Jakubczak

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2011, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Video broadcast and mobile video challenge the conventional wireless design. In broadcast and mobile scenarios the bit-rate supported by the channel differs across receivers and varies quickly over time. The conventional design however forces the source to pick a single bit-rate and degrades sharply when the channel cannot support it.

This thesis presents SoftCast, a clean-slate design for wireless video where the source transmits one video stream that each receiver decodes to a video quality commensurate with its specific instantaneous channel quality. To do so, SoftCast ensures the samples of the digital video signal transmitted on the channel are *linearly* related to the pixels' luminance. Thus, when channel noise perturbs the transmitted signal samples, the perturbation naturally translates into approximation in the original video pixels. Hence, a receiver with a good channel (low noise) obtains a high fidelity video, and a receiver with a bad channel (high noise) obtains a low fidelity video.

SoftCast's linear design in essence resembles the traditional analog approach to communication, which was abandoned in most major communication systems, as it does not enjoy the theoretical opimality of the digital separate design in point-to-point channels nor its effectiveness at compressing the source data. In this thesis, I show that in combination with decorrelating transforms common to modern digital video compression, the analog approach can achieve performance competitive with the prevalent digital design for a wide variety of practical point-to-point scenarios, and outperforms it in the broadcast and mobile scenarios.

Since the conventional bit-pipe interface of the wireless physical layer (PHY) forces the separation of source and channel coding, to realize SoftCast, architectural changes to the wireless PHY are necessary. This thesis discusses the design of RawPHY, a reorganization of the PHY which exposes a waveform interface to the channel while shielding the designers of the higher layers from much of the perplexity of the wireless channel.

I implement SoftCast and RawPHY using the GNURadio software and the USRP platform. Results from a 20-node testbed show that SoftCast improves the average

video quality (i.e., PSNR) across diverse broadcast receivers in our testbed by up to 5.5 dB in comparison to conventional single- or multi-layer video. Even for a single receiver, it eliminates video glitches caused by mobility and increases robustness to packet loss by an order of magnitude.

Thesis Supervisor: Dina Katabi
Title: Associate Professor

Acknowledgments

I am grateful to everyone who directly or indirectly contributed to completing this thesis. First and foremost, I thank Prof. Dina Katabi for years of advisorship and continued support throughout my graduate studies. Her encouragement was essential to my persistence on the topic. I thank my committee members: Prof. Hari Balakrishnan, Prof. Robert Morris and Prof. Vivek Goyal, for helpful discussions and their constructive criticism. I owe thanks to my collaborators: Hariharan Rahul, for help with the first physical prototype of SoftCast using the WARP software radio; and John Sun, for his contribution to the theoretical analysis of SoftCast in point-to-point settings. I thank Prof. Muriel Médard, Prof. Lizhong Zheng and Prof. Michelle Effros, who offered helpful advice at the early stage of the project.

I extend my thanks to my past collaborators: Prof. Dave Andersen, Dina Papagiannaki, Prof. Srinivasan Seshan, Michael Kaminsky, and Jay Kumar Sundararajan for much interesting research we conducted together. I am grateful to all of NMS, both present and past, for providing a supportive work atmosphere. A very special thanks to Mary McDavitt, whose heartwarming character brightened all administrative matters. I am much obliged to Prof. Terry Orlando for his advocacy.

Graduate school is not all work. I am glad to have met so many fascinating people at MIT, in CSAIL and in the Media Lab. I am lucky to be able to call some of them my friends.

To my parents for their ceaseless support and guiding me to the right path

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Video is already making up majority of Internet traffic [69,81], According to the Cisco visual networking index, *mobile* video will grow 66 fold over a period of five years [1]. Such predictions lead to a natural question: can existing wireless technologies, e.g., WiFi, WiMax, or LTE, support this impending demand and provide scalable and robust mobile video?

1.1 The Problem

Today, when a station transmits a wireless packet, it encodes the packet at a particular bit rate – a combination of error correction and digital modulation codes. At a high bit rate, the transmission is short and thus the station can send the data faster, but the packet is less resilient to noise. At a low bit rate, the encoding adds extra redundancy making the packet more resilient to noise, but reduces the net data rate. Thus, the bit rate choice presents a trade-off between throughput and robustness and it poses a challenge to scalable mobile video.

(a) Scalability. As demands for mobile video increase, congestion will also increase. The problem becomes particularly severe when many users try to watch a popular realtime event, e.g., the championship game of a national sport, such as Super Bowl. In such case, one would like to save bandwidth by multicasting the event as a single video stream. Different receivers, however, have different channel quality (i.e., signal-

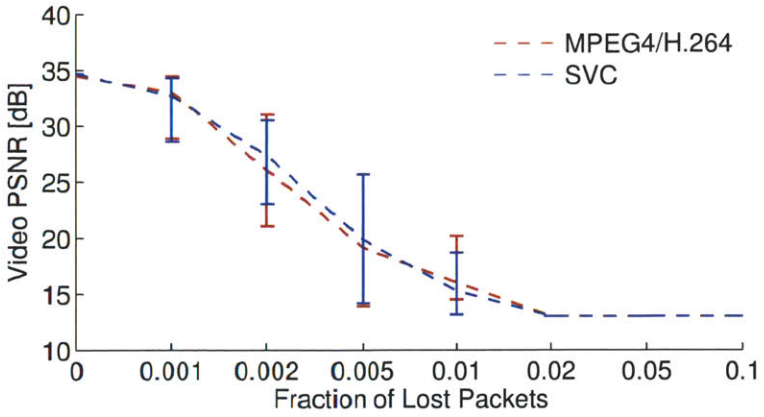


Figure 1-1: **Impact of interference-related packet loss on video quality.** PSNR below 20 dB corresponds to unacceptable video quality [79]. The figure shows that video encoded with either H.264/MPEG4 or SVC (i.e., layered) suffers dramatically at a packet loss rate as low as 1%.

to-noise ratio). Multicasting a single video stream to multiple receivers requires the source to transmit at the lowest bit rate supported by their channels. This reduces all receivers to the video quality of the receiver with the worst channel. Since such a design is undesirable from a user perspective, the typical approach today transmits an individual video stream to each receiver, even when all of these streams share the same content, which is unscalable.

(b) Robustness. The wireless medium suffers high bit error and packet loss rates due to both interference and channel noise. Video codecs however are very sensitive to errors and losses [4,89]. Fig. 1-1 plots the impact of interference-caused packet loss on MPEG4 (i.e., H.264/AVC) and SVC layered-video.[1] The figure is generated using the reference implementations of the two codecs [50,103], and by having an interferer transmit at regular intervals. (Other details are in §5.2.4.) The figure confirms past results [89], showing that both MPEG4 video and SVC layered video are highly sensitive to interference and become unviewable (i.e., PSNR $<$ 20 dB) when the packet loss rate is higher than 1%.

The lack of scalability and robustness in today's mobile video stems from the existing design of the network stack. Specifically, mobile video is impacted by two

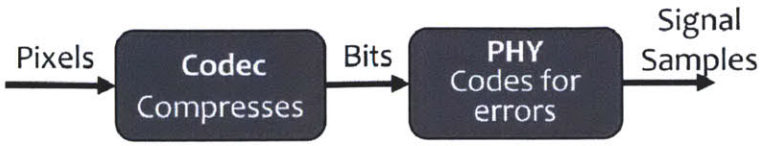
[1]SVC produces a base layer necessary for decoding and a refinement layer that adds details for receivers with better channels.

layers in the stack: the application video codec, which compresses the video, and the physical layer, which protects the video from channel errors and losses. Today, video codecs do an excellent job in compressing the video and removing redundancy. However, they also make the video highly vulnerable to bit errors and packet losses. In particular, all common video codecs use entropy coding (e.g., Huffman), in which a single bit flip can cause the receiver to confuse symbol boundaries, producing arbitrary errors in the video. This compressed video has to be transmitted over an erroneous wireless channel. Thus, the PHY layer has to add back redundancy in the form of error correction codes. Since the compressed video is highly fragile, video streaming requires the PHY to add excessive redundancy to eliminate the possibility of bit flips or packet loss. This approach is particularly inefficient in mobile video because the PHY needs to add excessive coding to deal with channel variations across time due to mobility or interference, and across space due to receiver diversity.

Theoretical results show that the existing layer separation – i.e., separating source coding (i.e., video compression) from channel coding (i.e., error protection) – is acceptable *only* in the case of unicast (point-to-point) channels and when the statistics of the channel are known *a priori* to the transmitter [87,96]. In practice, this means that, for a specific known channel SNR, one can find appropriate error correcting code to maximize the bit rate and guarantee sufficiently low error rate thus delivering optimal video performance. Such separation however becomes inefficient for multicast/broadcast channels, or when the channel's statistics are hard to predict due to mobility or interference [87,96].

1.2 Approach

SoftCast is a clean-slate end-to-end architecture for transmitting video over wireless channels. In contrast to the separate conventional design, SoftCast adopts a unified design that both encodes the video for compression and for error protection. This end-to-end approach enables multicast video delivery to multiple mobile receivers, with each receiver obtaining video quality commensurate with its specific instantaneous



(a) Conventional separate design

(b) SoftCast's joint design

Figure 1-2: **A comparison of SoftCast to the conventional wireless design.** The transmitter encodes video pixels to channel signal samples. In contrast to the conventional design, SoftCast replaces separately designed video compression codec and error protection codes with a unified linear transform.

channel quality.

SoftCast starts with video that is represented as a sequence of numbers, with each number representing a pixel luminance. Taking an end-to-end perspective, it then performs a sequence of transformations to obtain the final signal samples that are transmitted on the channel. The crucial property of SoftCast is that each transformation is linear. This property ensures that the signal samples transmitted on the channel are linearly related to the original pixel values. Thus, increasing channel noise progressively perturbs the transmitted bits in proportion to their significance for the video application, i.e., high-quality channels perturb only the least significant bits while low-quality channels still preserve the most significant bits. Each receiver therefore decodes the received signal into video whose quality is proportional to the quality of its specific instantaneous channel.

SoftCast's end-to-end architecture has the following four linear components:

(1) Compression: Traditional video compression is designed in separation from the wireless channel. Hence, though the wireless channel has a high error rate, traditional compression uses Huffman and differential encoding which are highly sensitive

to errors.[2] In contrast, SoftCast compresses a video by applying a three-dimensional decorrelation transform, such as 3D DCT. Using 3D DCT (as opposed to the 2D DCT used in MPEG), allows SoftCast to remove redundant information within a frame as well as across frames. Further, since DCT is linear, errors on the channel do not lead to disproportionate errors in the video.

(2) Error Protection: Traditional error protection codes may map values that are numerically far apart, *e.g.*, 2.5 and 0.3, to adjacent codewords, say, 01001000 and 01001001, causing a single bit flip to produce a dramatic change in the rendered video. In contrast, SoftCast's error protection is based on scaling the magnitude of the transmitted coded samples. Consider a channel that introduces an additive noise in the range ± 0.1. If a value of 2.5 is transmitted directly over this channel, it results in a received value in the range $[2.4 - 2.6]$. However, if the transmitter scales the value 10 times, the received signal varies between 24.9 and 25.1, and hence when scaled down to the original range, the received value is in the range $[2.51 - 2.49]$, and its best approximation given one decimal point is 2.5, which is the correct value. SoftCast has a built in optimization that identifies the proper scaling that minimizes video error subject to a given transmission power.

(3) Resilience to Packet Loss: Current video codecs employ differential encoding and motion compensation. These techniques create dependence between transmitted packets. As a result, the loss of one packet may cause subsequent correctly received packets to become undecodable. In contrast, SoftCast ensures that all packets contribute equally to the quality of the decoded video. Specifically, SoftCast employs a Hadamard transform [6] to distribute the video information across packets such that each packet has approximately the same amount of information.

(4) Transmission over OFDM: Modern wireless technologies (802.11, WiMax, Digital TV, etc.) use an OFDM-based physical layer (PHY). SoftCast is integrated

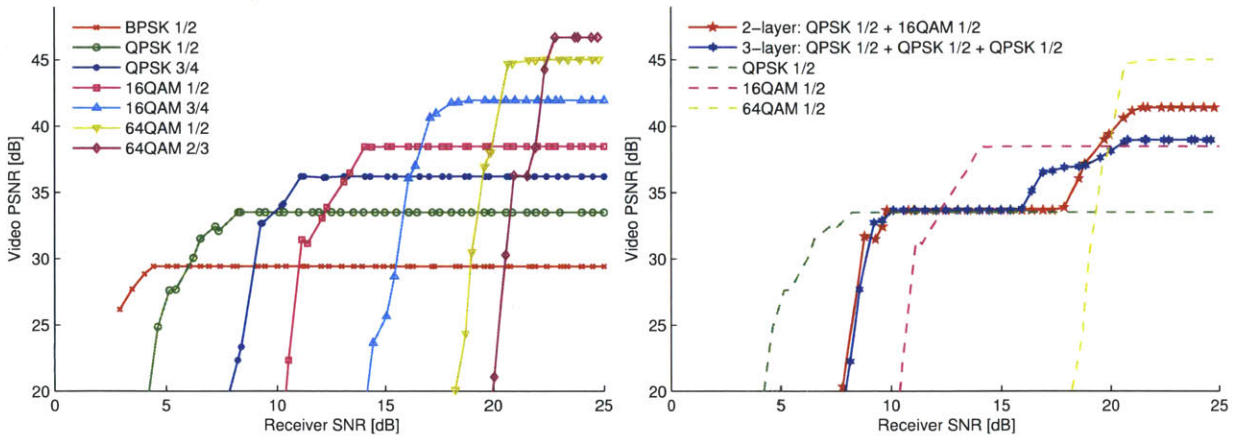
[2]Huffman is a variable length code and hence a bit error can cause the receiver to confuse symbol boundaries. Differential encoding and motion compensation encode frames with respect to other frames and hence any error in a reference frame percolates to other correctly received frames.

within the existing PHY layer by making OFDM transmit SoftCast's encoded data as the I and Q components of the digital signal.

SoftCast builds on prior work on video multicast over channels with varying quality. The state of the art approaches to this problem still use a separate design. These schemes use a layered approach in which the video is encoded into a low-quality base layer (which all receivers must correctly decode to obtain any video at all) and a few higher-quality enhancement layers (which receivers with higher-quality channels can decode to obtain higher-quality video). In the limit, as the number of layers becomes very large, a layered approach would ideally deliver to each receiver a video quality proportional to its channel quality. In practice, however, encoding video into layers incurs an overhead that accumulates with more layers [100]. Thus, practical layered schemes (such as those proposed for Digital TV) use only two layers [23,32,56]. In contrast to a layered approach, a SoftCast sender produces a single video stream, with the video quality at each receiver determined by the significance of the bits that its channel delivers without distortion. The quality of the video degrades smoothly at the granularity of the individual luminance bits, rather than at the much coarser granularity of the number of layers in the transmitted video. SoftCast also builds on a growing literature in information theory tackles joint source and channel coding (JSCC) [68,77,88]. SoftCast's design is motivated by the same philosophy but differs in its emphasis on linear transforms. Furthermore, past work on JSCC is mainly theoretical and is not tested over an actual wireless channel.

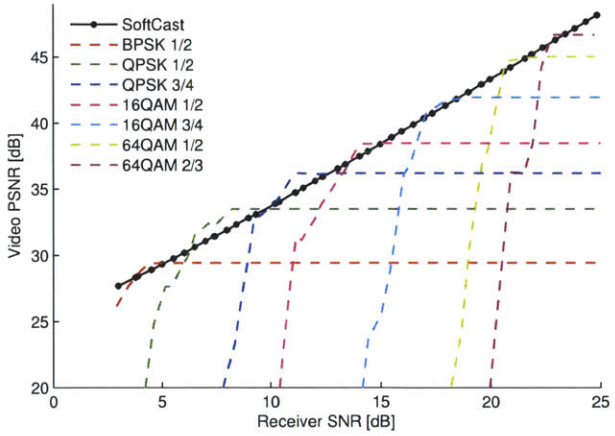
1.2.1 Graphical Comparison

Fig. 1-3 graphically displays the characteristics of the different video encoding and transmission schemes. This figure presents three graphs; each graph plots the video quality at the receiver as a function of the channel quality. All schemes use exactly the same transmission power and the same channel bandwidth over the same period of time, i.e., they are exposed to the same channel capacity and differences are due only to how effectively they use that capacity. The measurements are collected using GNURadio USRP nodes. For more details on the experimental setup see §5.1.



(a) Conventional Design

(b) Layered SVC with Hierarchical Modulation



(c) SoftCast

Figure 1-3: **Approaches to wireless video:** Fig. (a) plots the space of video qualities obtained with the conventional design which uses MPEG4 over 802.11. Each line refers to a choice of transmission bit rate (i.e., modulation and FEC). Fig. (a) shows that for any choice of bit rate, the conventional design experiences a performance cliff. Fig. (b) plots 2-layer video in red and 3-layer video in blue. For reference, the dashed lines are the three equivalent single-layer MPEG4 videos. The figure shows that layered video makes the cliffs milder but each new layer introduces overhead and reduces the maximum video quality. Fig. (c) shows SoftCast (in black) and single-layer MPEG4. It shows that SoftCast video quality fully scales with channel quality.

Fig. 1-3(a) illustrates the realizable space of video qualities for conventional MPEG-based approaches. Each line refers to a particular choice of transmission bit rate, i.e., a particular choice of forward error correction code and a modulation scheme. The video codec encodes the video at the same rate as the channel transmission bit rate. Fig. 1-3(a) shows that for any selection of transmission bit rate (i.e., modulation and FEC) the conventional design experiences a performance cliff, that is there is a critical SNR, below which the video is not watchable, and above that SNR the video quality does not improve with improvements in channel quality.

Fig. 1-3(b) illustrates the video qualities obtained by state of the art layered video coding. The video is encoded using the JSVM reference implementation for scalable video coding (SVC) [50]. The physical layer transmits the video using hierarchical modulation over OFDM, an inner convolutional code and an outer Reed-Solomon code following the recommendations in [23]. The figure shows two solid lines, the red line encodes the video into two layers while the blue line encodes the video into three layers. For reference, the figure also shows in dashed lines the single layer MPEG4 videos that span the range of channel SNRs spanned by the layers in the layered video. The figure shows that layered video transforms the performance cliff of the conventional design to a few milder cliffs. Layering however causes extra overhead [100] and thus increases the size of the video. Given a particular bit rate budget, the video codec has to reduce the quality of the layered video in comparison with the single layer video to ensure that the videos have the same size and can be streamed at the same bit rate. As a result, the enhancement layer of the 3-layer video has a lower quality than the corresponding layer in 2-layer video, which has a lower quality than the corresponding single layer video.

Fig. 1-3(c) illustrates the video qualities obtained with SoftCast. The figure shows that SoftCast's video quality is proportional to the channel quality and stays competitive with the envelope of all of MPEG curves.

1.3 Contributions

This thesis makes the following contributions.

- It presents SoftCast, a novel design for wireless video, where the sender need not know the wireless channel quality or adapt to it. Still, the sender can broadcast a video stream that each receiver decodes to a video whose quality is commensurate with its channel quality. This happens without receiver feedback, bit rate adaption, or video code rate adaptation.

- SoftCast includes a packet loss protection scheme compatible with the degradable nature of the video content. Unlike existing video approaches where some packets are more important than others, SoftCast protects the data from loss of whole packets by ensuring that all packets contribute equally to the quality of the decoded video. The distortion resulting from packet loss is then proportional to the number of lost packets, rather than dependent on which packets have been lost. This property significantly increases resilience to packet loss.

- It discusses RawPHY, a new architecture for the wireless PHY layer that enables flexible error protection for wireless applications, and in particular integration of SoftCast. RawPHY shields the designers of the higher layers from much of the perplexity of the wireless channel while exposing a waveform – rather than binary – interface to the channel.

- It presents an implementation and an empirical evaluation of SoftCast and RawPHY in a 20-node testbed of software radios. It shows that the protocol significantly improves robustness to mobility and packet loss and provides a better quality video multicast.

- It includes information-theoretic analysis of the performance bounds of SoftCast's analog design to the conventional digital design that separates source and channel coding by one or more layers of bits. It shows that for multivariate correlated Gaussian sources, there exist regimes where the analog scheme

is near-optimal for stationary point-to-point communication and provides significant gains over the best theoretical performance of a digital design for non-stationary (broadcast or mobile) scenarios.

1.3.1 Summary of Experimental Results

I have implemented SoftCast and evaluated it in a testbed of 20 GNURadio USRP nodes [33,94]. I compare SoftCast with two baselines: 1) MPEG-4 (i.e., H.264/AVC) over 802.11, and 2) layered video where the layers are encoded using the scalable video extension to H.264 (SVC) and transmitted using hierarchical modulation as in [56]. I evaluate these schemes using the Peak Signal-to-Noise Ratio (PSNR), a standard metric of video quality [65,79]. I have the following findings:

- SoftCast delivers to each multicast receiver a video quality that is proportional to its channel quality and is competitive (within 1 dB) with the optimal quality the receiver could obtain if it were the only receiver in the multicast group.
- For multicast receivers of SNRs in the range $[5, 25]$ dB, SoftCast improves the average video quality by 5.5 dB over the best performer of the two baselines.
- Even with a single mobile receiver, SoftCast eliminates video glitches, whereas 14% of the frames in our mobility experiments suffer glitches with the best performer of the two baselines.
- Finally, SoftCast tolerates an order of magnitude higher packet loss rates than both baselines.

1.4 Organization

Chapter 2 offers a quick primer on the fundamentals of source and channel coding theory as well as practical considerations in wireless communication systems and video compression. Chapter 3 presents the design of SoftCast's linear code for video compression and error protection. Chapter 4 presents RawOFDM, a realization of

RawPHY which allows to integrate SoftCast in the network stack without having to redesign the whole PHY. Chapter 5 includes experimental results from a 20-node testbed as well as the theoretical analysis.

1.4.1 Previously Published Material

I have presented a preliminary design of SoftCast in [46]. This design was extended to include inter-frame coding and presented in [43]. The results of the complete design as included in Chapter 3 have been published in [45]. The prototype design of RawOFDM was described in [46] and a GNURadio implementation of both RawOFDM and SoftCast was presented in a real-time demonstration in [44]. The theoretical results in Chapter 5 have been published in [47].

Chapter 2

Background

Before I describe the design and implementation of SoftCast, I provide the reader with a quick primer on the fundamentals of source and channel coding theory as well as some practical considerations in wireless communication systems and video compression. This chapter offers tools that will aid understanding of the rest of this thesis. I limit the exposition to the major concepts rather than the formal definition and proofs. Details and excellent textbook treatment for the core information-theoretic results can be found in [16,62].

Source and Channel Coding: In his 1948 paper, Claude Shannon introduced the notion of *channel capacity*, the tightest upper bound on the amount of information that can be reliably transmitted over a communications channel [86]. Implicit in his seminal theorem is that the communication effort can be split between a source coder and a channel coder, where the source coder deals only with the statistics of the ran-

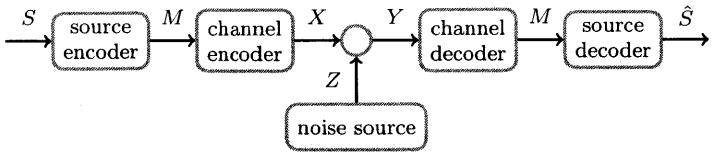


Figure 2-1: **Source and channel coding.** By separate source-channel design, the output of the channel decoder is an error-free replica of the binary message M generated by the source encoder, although the overall output of the system $\hat{S}$ is generally an approximation of the original source signal S due to compression loss.

dom source and the channel coder deals only with the statistics of the random channel. As illustrated in Fig. 2-1, between the source and channel coders flows a memoryless, uniformly-random binary stream at some fixed rate (bit/s). Although, in this thesis, I discuss the limitations of such separation, I will leverage many techniques of source and channel coding which will also be used as the reference baseline in evaluation. I end this chapter with the introduction of most common joint source-channel coding (JSCC) techniques that do not abide by this separation but can improve performance in scenarios when such separation is suboptimal.

2.1 Source Coding

In the context of data communications, the term *source coding* refers to reduction in the bit-rate necessary to convey the source information either without loss (lossless compression) or with a distortion criterion (lossy compression). In theory, the bit-stream to be decoded is exactly as it was produced by the encoder and any information loss is caused by the encoder. In practice, many compression algorithms have some built-in mechanisms to detect (and conceal) a small number of errors, which inevitably comes at the price of compression efficiency. We will discuss some of these techniques in the context of video compression.

Notably, in the context of networks, source coding could refer to any coding applied at the source node, including additional redundancy to protect from loss within the network. However, in this thesis, we will use it strictly in the context of lossy data compression.

2.1.1 Rate-Distortion Theory

If some loss of information is tolerable, the mismatch between the original data and the decoded reconstruction is measured by *distortion*, often denoted by D. Naturally, we are interested in the lowest distortion achievable at a particular bit-rate, R, or conversely the lowest bit-rate required to achieve particular distortion. This trade-off can be captured by curves $D(R)$ and $R(D)$, referred to as rate-distortion curves.

Rate-distortion theory deals with finding rate-distortion curves for specific source models and distortion curves [16].

Formally, the source produces a random variable X, the encoder computes a function $f(X^n) \in \{1, 2, \ldots 2^{nR}\}$, i.e., maps n source symbols to one of 2^{nR} indices, which require nR bits to represent. Hence the average bit-rate is R with block-length of n. The decoder reconstructs $\hat{X}^n$. The non-negative distortion function is defined for a source symbol x and its reconstruction $\hat{x}$:

$$d(x, \hat{x}) \in R^+$$

with the requirement that $d(x, x) = 0$. The distortion between two sequences is defined as

$$d(x^n, \hat{x}^n) = \frac{1}{n} \sum_{i=0}^{n} d(x_i, \hat{x}_i)$$

We are interested in the expected distortion

$$D = E[d(x^n, \hat{x}^n)]$$

where the expectation is with respect to the probability distribution of X. The rate-distortion function $R(D)$ is defined as the infimum of rates R such that there exists a sequence of codes of bit-rate R whose expected distortion is less than D as the block-length n tends to infinity. I.e., it defines the fundamental bound of the rate-distortion trade-off regardless of how many source symbols are coded together.

Memoryless Gaussian source with MSE distortion: A well-treated example is the rate-distortion function of a random iid variable with distribution $N(0, \sigma^2)$ under the mean-square-error (MSE) distortion metric $d(x, \hat{x}) = (x - \hat{x})^2$ [16]. The minimum rate required to achieve expected distortion D is

$$R(D) = \frac{1}{2} \log_2 \left(\frac{\sigma^2}{D} \right)$$

Note that for fixed rate, distortion is directly proportional to variance, so it is customary to consider source-to-distortion ratio (SDR) defined as:

$$SDR = \frac{\sigma^2}{D}$$

Hence the rule of thumb that *6dB in SDR requires 1 bit in rate.*

Remarks: Under the MSE distortion metric, the memoryless Gaussian source is the hardest to encode, i.e. all other sources with the same variance have $R(D)$ below the Gaussian rate-distortion curve [16]. Furthermore, if the source is not memoryless, but rather exhibits some auto-correlation, it will require lower rate for the same SDR than a memoryless source. This hinges on the assumption that the encoder/decoder know the conditional probability distribution of the source, but that prior knowledge reduces the rate required to describe the sequence of source symbols. Along the same lines, a multi-variate source (where each symbol is a vector in R^n) in which the dimensions are independent but some have lower variance than others, will require lower rate (per source symbol) than if all source dimensions were iid (white). Intuitively, the lower-variance dimensions require lower rate bringing the overall rate down. Generally, the more "detailed" the model of the source the more compression potential we have. In practice, many dynamic parameters of such model, such as frequency of source symbols, variance or covariance of the joint probability distribution, etc., can be learned by the encoder and communicated to the decoder in-band (as metadata), although the rate required to encode this description cannot be ignored.

2.1.2 Compression Techniques

Let us introduce some of the most popular compression techniques. We specifically focus on techniques that are commonly used in image and video compression, but introduce the general ideas first, before discussing their application in that domain.

symbol	probability	codeword
a	0.8	0
b	0.1	10
c	0.05	110
d	0.05	111

Table 2.1: **Entropy coding.** Example source probability distribution over an alphabet of four symbols and one possible binary source code.

Entropy Coding: For a source with a finite alphabet of size 2^k but non-uniform distribution over the alphabet or non-zero auto-correlation, it is possible to use fewer bits than k per symbol to encode it *losslessly*.

Consider the source with probability distribution over the alphabet $\{a, b, c, d\}$ presented in Table 2.1. Rather than using 2 bits per symbol $\{00, 01, 10, 11\}$ we could use the codewords shown in the table and achieve an average bit-rate of 1.3 bits/symbol. Observe that in this variable-length code, each symbol is mapped to one codeword. In general, a codeword could encode more than one symbol or a variable number of symbols. Furthermore, this *prefix code* has the property that no codeword is a prefix of another, and hence can be decoded without ambiguity via simple lookup. Simplicity of the decoder is a common property in many popular source coding techniques, but is not a requirement.

The reduction in bit-rate comes from the arrangement that more popular symbols are assigned shorter codewords and vice-versa. Shannon has shown that the optimal bit-rate for a source with distribution $p_X(x)$ over its alphabet $\{x_1, x_2, \ldots\}$ equals its entropy defined as:

$$H(X) = E\left[-\log_2 p_X(x)\right] = \sum_i -p_X(x_i)\log_2 p_X(x_i)$$

Observe that for a given alphabet size N, the entropy is highest when the distribution is uniform over all elements and equals $\log_2 N$. Looking back at Table 2.1, the entropy of the source is around 1.022. The achieved rate of 1.3 bits/symbol is actually the best possible for any code with block-length 1, i.e., encoding each symbol independently. To compress further, one would consider blocks of 2 or more symbols. For instance

the optimal code of block-length 2 achieves 1.06 bits/symbol. As the block-length goes to infinity, the bit-rate will approach the entropy, but with diminishing returns.

Most well-known entropy coding techniques are Huffman and arithmetic coding [40,62]. The code in Table 2.1 is actually produced by the Huffman algorithm, which constructs the codebook by iteratively replacing the two least likely symbols by a new symbol and using one bit to distinguish between them. The Huffman algorithm achieves the optimal bit-rate for given block-length. Arithmetic coding differs from Huffman coding in that rather than mapping each input symbol to a codeword separately, it encodes the entire message into a single number, a fraction between 0 and 1, represented by its binary expansion. The more likely source symbols are mapped to wider sub-intervals of the current interval. When the end of input is reached, a single number from the remaining interval is selected, such with the shortest binary expansion. Huffman coding can be considered as a special case of arithmetic coding with fixed block-length, and is often less efficient when the source probabilities are not round binary fractions.

Note that a source with uniform distribution over its alphabet of size 2^k would not be compressible at all, as its entropy is k. However, if such a source is not memoryless, then sequences of symbols are not uniformly distributed and can be compressed. For instance, consider a source which reports the current state of a switch that flips rarely. Either state of the switch (0 or 1) is equally probable, so the entropy with block-length 1 is 1 bit/symbol, but given the current report, we would bet that the next report will be the same. *Run-length encoding* (RLE) is a very simple method which replaces sequences of the same symbol (or *runs*) with a single value and its repetition count. In this case, a sequence 0000011111110000111 could be represented as $(0,5)(1,7)(0,4)(1,3)$ or actually $5,7,4,3$ and could be encoded as 101111100011 (using 3 bits per run) to reduce the size from 19 to 12 bits. RLE, however, is very limited. Suppose the source reports the current temperature. We expect the temperature to change slowly, but if it differs from the previous value by even one degree, RLE would need to encode a new run. A general entropy coder with longer block-length could consider the probability of longer sequences and determine that sequences changing

slowly are less probable. In practice, a decorrelation step is performed to "convert" a source that is uniformly-distributed but with autocorrelation to a source that shows little autocorrelation but is distributed non-uniformly, and thus has lower unit-block-length entropy. We will discuss decorrelation techniques such as transform coding and predictive coding later in this section, but first we discuss lossy compression.

Quantization: In many cases, perfect reconstruction is not necessary and some amount of information loss is tolerable. In particular, content for consumption by human senses falls into that category as human perception is limited. Similarly, signals from analog sensors and measurements are typically unnecessarily precise given the engineering necessity or instrument noise. To reduce the bit-rate, the source encoder can choose to discard some of the precision and encode a lower resolution signal. In case of a digital signal this is commonly done by *subsampling*, i.e., discarding some source symbols completely[1], or *quantization*, that is a non-injective mapping of source alphabet to the codebook, i.e. assigning one codeword to multiple elements of the alphabet. For example, one could reduce the bit-rate of a source of 16-bit numbers by simply discarding or rounding off the least significant 8 bits, thus mapping 256 source values into each codeword. Quantization can also be used more generally to refer to the process of mapping a continuous range of values to a finite number of discrete levels, e.g., as part of the analog-to-digital conversion. The goal is to *approximate* the original alphabet by a codebook of smaller cardinality.

In the rate-distortion trade-off, the more quantization levels, the lower the quantization error (the expected error between the original and reconstruction), but the higher the required rate. In principle, the encoder assigns codewords to non-overlapping ranges of source values and the decoder maps the received codewords to reconstructed values. Hence, a quantizer is typically defined by decision boundaries specifying the input ranges and reconstruction levels specifying the decoded values corresponding to each range. The quantizer codewords can be simply compressed using entropy coding described above. This would often be the case if the source probability density

[1]Notably, subsampling needs not be lossy, for instance, if the signal is band-limited below the sampling frequency after subsampling.

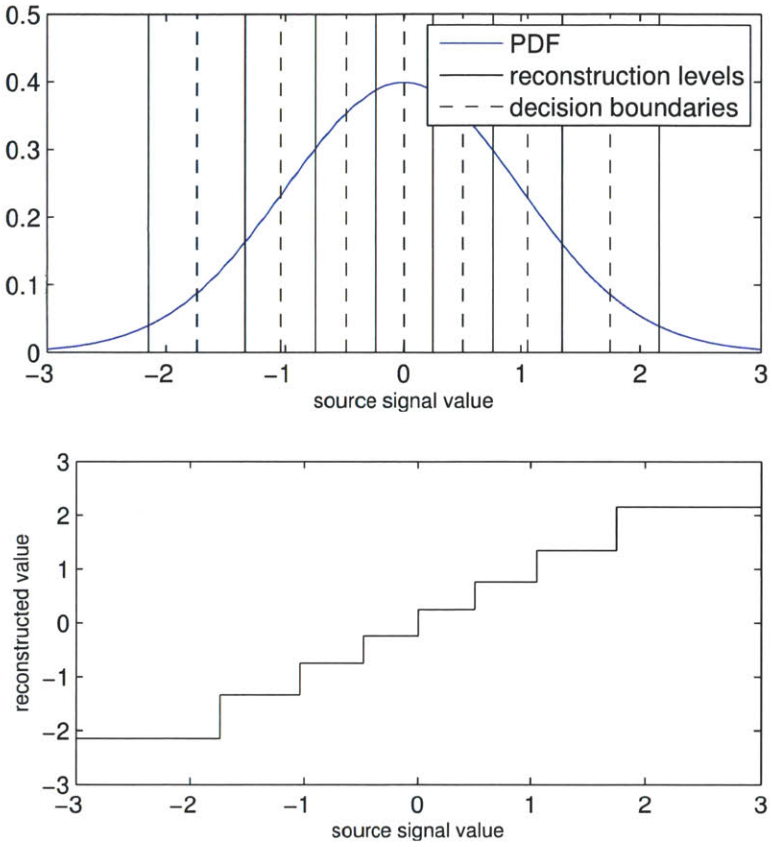


Figure 2-2: **Principle of quantization.** A quantizer is a non-injective function which maps the continuous source alphabet to a finite codebook. A quantizer assigns codewords to disjoint ranges of source values and determines the decoded symbols corresponding to each codeword. Transmitted over the channel is just the index of the codeword. Shown here is (a) an example probability distribution function for a Gaussian source with the decision boundaries and reconstruction levels of an 8-level (3-bit) Lloyd-Max quantizer [60,66], and (b) a plot of the encoding function.

function is non-uniform but the input ranges have all equal length.

For fixed-length codewords one can do better by allowing non-uniform quantization ranges. The goal then is to minimize the distortion given fixed number of levels. For MSE distortion, the conditions on the optimal quantizer, found by Lloyd and Max, require that the decision boundaries are the mid-points between reconstruction levels and the reconstruction levels are the centroids of the probability distribution function within each range between the decision boundaries [60,66]. This recursive definition requires an iterative algorithm, known as the Lloyd-Max algorithm.

As in the case of lossless compression, where the optimal codebook with block-length 1 did not achieve entropy, the optimal scalar quantizer (with block-length 1) does not achieve the rate-distortion function. A *vector quantizer* maps vectors of input symbols to codewords. Rather than scalar ranges, decision boundaries define cells in the multidimensional space. Vector quantization is optimal in the limit of infinite block-length, and can deal with sources with auto-correlation, but encoding is computationally expensive (even with pre-optimized decision boundaries) due to dimensionality explosion. Hence, practical methods apply decorrelation via transform or predictive coding therefore performing scalar quantization.

Transform Coding: When there is auto-correlation in the source signal, encoding more than one source symbol becomes essential (as the average entropy of a sequence is much lower than the entropy of each symbol). Likewise, when the source signal has multiple components exhibiting cross-correlation. General block-n encoders (such as Huffman, arithmetic or Lloyd-Max quantizer) are complex, since the alphabet / signal space grows exponentially with block length. In contrast, if the source can be represented by a vector of independent random variables (or bands), then each dimension in such vector can be encoded independently without substantial loss in performance. For now we focus on the case when components of the source signal are dependent linearly.

Converting a vector of correlated random variables into a vector of orthogonal (i.e., linearly independent or uncorrelated) variables can be done universally by means of

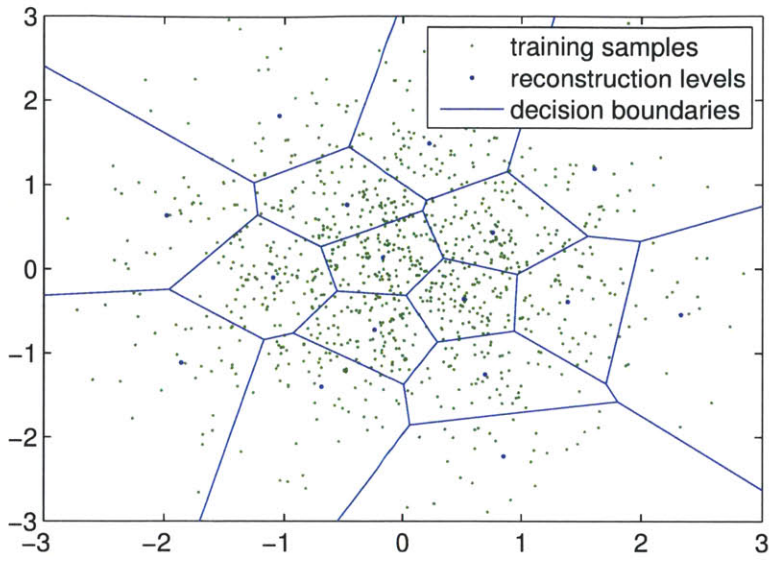


Figure 2-3: **Vector quantization.** A vector quantizer assigns codewords to regions of the input space. The decision boundaries in the optimal quantizer are equidistant from the reconstruction vectors. Shown in figure is an example of a 2-d vector quantizer found using the Linde-Buzo-Gray algorithm [59]. The decision boundaries form a Voronoi diagram on the plane.

the Karhunen-Loéve transform (KLT) [61]. The Karhunen-Loéve theorem applies to general stochastic process not just random vectors, but for random vectors it is equivalent to singular value decomposition (SVD) of the covariance matrix.

Specifically, consider a random vector X with covariance matrix Λ_X. Then we can decompose:

$$\Lambda_X = U\Sigma U^T$$

so that U is an orthogonal matrix, ^{T} is transpose and Σ is a diagonal matrix of non-negative values. The rows of U are eigenvectors while the corresponding elements of Σ are the corresponding eigenvalues of Λ_X. Now consider the random vector $Y = U^T X$. Its covariance matrix is

$$\Lambda_Y = U^T \Lambda_X U = U^T U \Sigma U^T U = \Sigma$$

which is diagonal. Hence, the components of Y are orthogonal.

After the linear transform, the decorrelated signal is further processed by quanti-

zation (in lossy compression) or directly by entropy coding (in lossless compression). At the decoder, the transform is inverted $X = U\hat{Y}$ on the dequantized signal $\hat{Y} \approx Y$.

The downside of the SVD method is that the transform U, which is necessary to decode, is dependent on the covariance matrix Λ_X. Communicating the values in either of these matrices might require substantial out-of-band capacity and would grow with square of the vector size. The alternative is to use a preset transform matrix. In that case, the operation is essentially a projection of the source signal onto a new basis, with the assumption that the signal is "sparse" in that basis. Some of the most widely used transforms are the Fourier transform, the closely related discrete cosine transform (DCT), the Hadamard transform and the wavelet transform. Their common feature is energy companding[2], i.e., after transform, most of the source components have a low variance, and conversely the energy is concentrated in a few components. Closely related to transform coding are *subband coders* which use wavelet or other filter banks (often hierarchical) to decompose the signal into uncorrelated subbands and then code each sub-band independently [18].

Predictive Coding: Another approach to exploiting auto-correlation or cross-correlation in the source signal is to use some of its elements to predict the others according to an assumed model of source. For instance, a strongly auto-correlated signal could be extrapolated from the past history or interpolated using samples from both past and the future. In either case, rather than a sequence of direct signal samples, the encoder emits: the reference samples, parameters of the prediction model, and the difference between the prediction and the actual value, i.e., the *residual.* Predictive coding is often recursive, i.e., the predicted value can itself be used to predict other values. If recursive predictive coding is followed by lossy compression, it is critical that the decoded version of the reference values is used for prediction, since the lossless version will not be available for reconstruction at the decoder.

An example of one of the simplest predictive coding schemes is DPCM (differential pulse-code modulation) which uses the previous value as prediction for the next

[2]on the specific class of sources that they are applied to

value. In that case, the model is non-adaptive and the predictive coder is essentially equivalent to applying a linear transform, with the difference that it operates on an infinite stream rather than a finite size block. In practice, a reference value is used periodically to allow recovery from decoding errors or missing data. In general, the predictive model can be non-linear and/or adaptive. For instance, an encoder could use a subset of past samples for prediction and encode the indices of those samples in parameters of the model.

2.1.3 Video Compression Primer

Now that we introduced the basic concepts of source coding, we provide a quick primer on the anatomy of modern video coders. Video compression exploits redundancy within the video data:

spatial: pixels in the same frame are highly correlated. In natural images many areas have slowly changing luminance and colors.

temporal: adjacent frames in a sequence are similar. A static background is the same across a sequence. An object in motion or a scene during camera pan is composed of the same pixels, but at a changing offset.

psycho-visual: sensitivity of human vision is selective. Human eye is more sensitive to changes in luminance than chroma (color). It has also limited acuity for fine detail, in particular over slow changes in luminance and chroma, i.e. a shifted edge is much more detectable than a shift in a gradient.

The spatial and temporal redundancy is exploited by transform and predictive coding, while the psycho-visual effects are taken into account in quantization.

Figure 2-4 presents a block diagram of the state-of-the-art video encoder: joint ITU/MPEG standard H.264/AVC (Advanced Video Coding) or MPEG-4 part 10.

Color Transform and Subsampling: The most popular way of representing color in computer systems is the RGB model founded on the *trichromacy* of human vision.

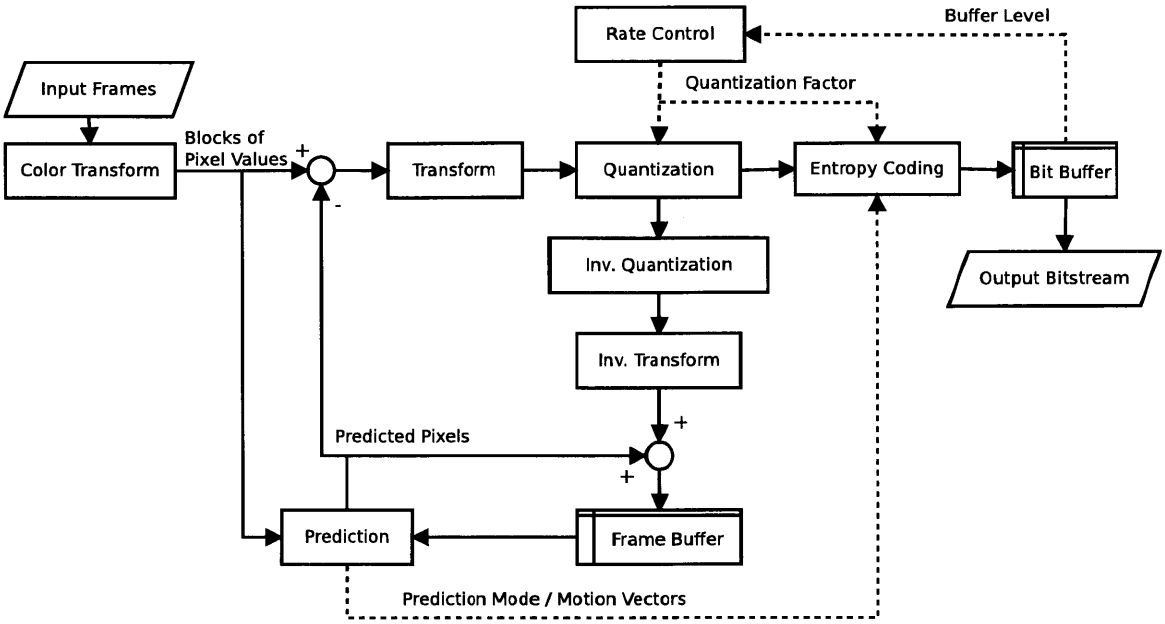


Figure 2-4: **Anatomy of a modern video coder.** H.264/AVC, i.e., MPEG-4 part 10. The *Prediction* block includes intra-frame prediction and motion-compensated inter-frame prediction.

Thus, the input data has three channels: red, green and blue. However, encoding each of the channels independently would be inefficient, as there is high correlation in the channels and human eye is more sensitive to luminance than color. Thus, the first step of most video compression algorithms is to perform a color transform, i.e. project the three-dimensional pixel values onto a different color space. The YUV (and closely related Y'CbCr) color space uses the Y dimension to represent luminance and the UV dimensions to encode color. As the human eye is less sensitive to variations in color than luminance, the data in the chroma channels is subsequently *subsampled*, usually at a rate of 2. For example, the typical 4:2:0 format represents a 16×16 macroblock of pixels by a 16×16 Y block, 8×8 U block and a 8×8 V block, thus requiring half the number of values needed before subsampling.

Predictive Coding: The coder operates on macroblocks of 16×16 pixels, but they are not encoded independently. Rather a set of previously encoded macroblocks can by used for to compute a prediction of the block to be encoded. This prediction is then subtracted from the original block and only the difference is encoded. It is important

to note that the *reference* data used for prediction must be available at the decoder for reconstruction. Therefore, the frame buffer stores pixel values as they would be available at the decoder, i.e., after lossy compression. Furthermore, note that the decoder cannot decode predicted pixels until the reference data is reconstructed, which introduces constraints on the ordering of data in the bitstream.

Intra-frame predictive coding uses for reference only the pixels within the frame. H.264 defines 9 modes – directions of prediction – including horizontal, vertical, diagonal, etc. I.e., in horizontal prediction, the pixels are predicted to match the values in their direct left neighbors.

Inter-frame coding uses as reference pixels from surrounding frames in the sequence. To exploit similarity in pixels despite motion of the object or the whole scene (caused by panning), the reference does not need to correspond to the same spatial coordinates, but rather be offset by a vector. Thus, for each macroblock to be predicted, the coder estimates a *motion vector* to its best reference. For better efficiency, more than one reference could be used, including the frames that follow in the temporal order. Furthermore, the references can be weighted in a linear combination. This is particularly useful when encoding fading and cross-fading scenes.

In MPEG-2, each frame has a particular reference. I-frames are not predicted. P-frames are predicted using the preceding I- or P-frame. Finally, B-frames are predicted using both the directly preceding and following I- or P-frames. In H.264, the I/P/B distinction can be made per macroblock, more than 2 (up to 16) weighted reference frames can be used, B-frames can be used for prediction (in a so called B-frame hierarchy), and motion can be estimated independently for parts (as small as 4×4) of the macroblock.

It is important to note that the intra-frame prediction mode, as well as the inter-frame prediction type, motion vectors and weights, are parameters of the predictive model and need to be encoded in the bit-stream. Motion vectors, for instance, are often correlated within a frame, and therefore encoded differentially in another application of predictive coding.

As a final note, predictive coding, motion estimation in particular, is the most

computationally intensive step in the modern video coder which heavily skews the computational effort towards the encoder side, which has to perform the search for the best mode and motion vectors, while the decoder side performs a simple lookup.

Transform: As pixels within a macroblock are highly correlated, the encoder applies a decorrelating transform to compand the energy of the signal down to a few components. The most prevalent is the 8×8 2-dimensional DCT (Discrete Cosine Transform) applied within each block, but other transforms (notably wavelets and Hadamard) have been used in some video and image coders.

Adaptive Quantization: The DCT coefficients are subsequently quantized, by simple divide-and-truncate operation. The greater the divisor, the more coarse the approximation. The divisor might be different for each DCT coefficient in accordance with the human psycho-visual model. Typically, the coefficients corresponding to high spatial frequencies are quantized more coarsely than the low spatial frequencies, and DC (i.e., the average value in the whole frame) is quantized most finely. That said, when inter-frame prediction is used, the residual after prediction does not resemble natural images and is typically quantized with a uniform quant size.

The quantizer step is multiplied by the *quantization factor* which lets the video coder trade rate off for distortion. The coarser the approximation, the lower the decoded video quality, but also the fewer bits are required to encode it. In the streaming scenario, the encoder must keep the bit-rate constant and so, when the buffer is reaching its limit, it needs to increase the quantization factor thus reducing quality and rate. In practice, the rate control process is more complex, as it is difficult to predict the rate that will result in using a specific quantization factor.

Entropy Coding: The quantized coefficients, as well as all the parameters such as: prediction mode, motion vectors and weights, quantization factors, etc., are then compressed without loss in the entropy coder. The entropy coder might user precomputed codebook, but could also be adaptive in which case the codebook needs to be encoded as well.

Legacy video coders (MPEG-2 and its foundation, JPEG) used a combination of a Huffman variable-length code (VLC) and run-length encoding (RLE) in which a non-zero DCT coefficient followed by a run of zero-valued coefficients would be encoded as one codeword. To facilitate that, the coefficients would be encoded following a special zig-zag order. The modern H.264 coder uses more sophisticated context-adaptive VLC or arithmetic coding (CAVLC or CABAC).

2.2 Channel Coding

While source coding reduces the bit-rate by removing redundancy from the bit stream, channel coding creates structured redundancy in the transmitted symbols in order to help the decoder invert the destructive effects of the channel.

2.2.1 Channel Model and Capacity

The channel coder takes a sequence of bits and maps it to a codeword which is sent on the channel. The decoder then must infer the original sequence of bits from the distorted codeword it received. Just as source coders are designed around a statistical model for the source, channel coders address the channel model.

In general, a memoryless channel model is captured by the conditional distribution $p_{Y|X}(y|x)$ where X is the channel input symbol and Y is the channel output symbol. In 1948, Claude Shannon defined *channel capacity* as the tightest upper bound on the amount of information that can be reliably transmitted over the channel, given its probabilistic model. Capacity is given by:

$$C = \sup_{p_X} I(X;Y)$$

where $I(X;Y)$ is the *mutual information* between X and Y defined as

$$I(X;Y) = E\left[\log_2 \frac{p_{X,Y}(x,y)}{p_X(x), p_Y(y)}\right]$$

where $p_{X,Y}$ is the joint distribution and p_X, p_Y are the respective marginal distributions. Equivalently, using the definition of entropy from section 2.1.2:

$$I(X;Y) = H(X) + H(Y) - H(X,Y)$$

Note that p_X is determined by the designer of the channel code, i.e., codebook. Shannon has shown that for a given channel with capacity C, and information transmitted at rate R, then there exist channel codes that allow the probability of error at the receiver to be made arbitrarily small *if and only if* $R < C$. As in rate-distortion theory, the practical limit to achieving capacity is the block-length of the channel coder, i.e., number of channel symbols spanned by each codeword, which determines delay and complexity of both encoder and decoder.

Commonly considered *memoryless* models and their capacities are:

additive white Gaussian noise (AWGN) channel: adds a random variable (noise) with normal distribution, $N(0, \sigma^2)$ to the input value.

$$C = \frac{1}{2}\log_2(1 + SNR)$$

where the *signal-to-noise ratio*, $SNR = E[X^2]/\sigma^2$

binary symmetric channel (BSC): flips the input bit (0 to 1 and vice versa) with fixed probability, p.

$$C = 1 - H(p)$$

where $H(p)$ is the entropy of a Bernoulli trial with probability of success p, i.e. $H(p) = -p\log_2 p - (1-p)\log_2(1-p)$.

binary erasure channel (BEC): replaces input with ϵ ("erased") with fixed probability, p.

$$C = 1 - p$$

Wireless Channel: The above models do not directly apply to the wireless channel. Typical wireless applications utilize RF frequencies in a passband, which allows multiple concurrent transmission in different frequency ranges (commonly referred to as a different channels) and efficient propagation through the environment. However, all or most of the digital processing is done at the *baseband*, i.e. before analog interpolation and up-conversion at the transmitter and after down-conversion and sampling at the receiver. Therefore from a communication system design point of view, it is most useful to consider the *discrete-time* baseband model of the wireless channel. In principle, a (stationary) wireless channel which uses radio frequency EM waves can be modeled as a linear time-invariant (LTI) system. Thus, it can be characterized by its impulse or frequency response, determined by electrical properties of the environment and the communications hardware. In practice, the response is unknown and can change over time. Additionally, the signal can experience non-linear and time-dependent distortion during the digital and analog processing at the transmitter or receiver, or due to Doppler shift. Furthermore, signals from other sources, such as interference or circuit noise, can add up and distort the information signal at the receiver. Finally, a defining characteristic of the mobile wireless channel is *fading*, i.e., the variations of the channel quality over time, space and frequency.

2.2.2 Channel Coding Techniques

In this section, we briefly describe a series of techniques that together will wrap the wireless channel into a lossless bitpipe. The modern wireless communication chain applies each of the above models at some point in the pipeline. We introduce the general ideas first, before discussing their application in the coded OFDM system.

Training: The impulse response (or equivalently the phase and amplitude of the frequency response) of the wireless channel is determined by the propagation environment, in particular, by obstacles which attenuate or reflect the radio waves. A frequency shift caused by Doppler effect or oscillator mismatch between the transmitter and receiver devices is also initially unknown to the communicating parties. In

order to effectively deal with this distortion (primarily by compensating for it at the receiver), the transmitter includes known symbols that allow the receiver to estimate parameters for the channel model. These known symbols can be included at the beginning of the transmission, e.g., as the *preamble* of the packet, or intermingled with data, e.g. *pilots*. In many systems, training is essential not only to compensate for channel distortion but also to find the beginning of the packet and establish *timing*, i.e., when does one symbol end and another begin.

Equalization and Orthogonalization: As the EM wave propagates through environment, it reflects from surfaces and may reach the receiver antenna via multiple different paths. As multiple copies of the signal add up with different delays and attenuations, the impulse response of the channel contains multiple copies of the impulse. Conversely, some frequencies within the signal will add up destructively and so the frequency response is no longer flat. Consequently, a symbol transmitted on the channel is spread in time and can interfere with subsequent symbols. There are two major ways to combat this *inter-symbol interference* (ISI): equalization and orthogonalization.

Equalization attempts to counter the effects of ISI at the receiver by estimating channel response (via training) and compensating for it, e.g., by decoding the previous symbol, recoding it, and canceling its interference, or by filtering the signal to make the frequency response flat.

Orthogonalization avoids ISI by ensuring that two symbols cannot interfere with each other. A *guard interval* inserted between symbols, if longer than the delay spread of the channel, ensures that there will be no ISI. A *pulse-shaping* filter at the transmitter can ensure that the channel satisfies the Nyquist ISI criterion, i.e., that the impulse response is zero at subsequent symbol sampling points, so it does not create ISI despite spreading over multiple symbols.

Orthogonalization in Frequency Domain: Alternatively, if we consider a signal of length T but in the frequency domain, then we can see that frequencies at multiplies

of $1/T$ are orthogonal[3] and can be used to encode symbols which will not interfere with each other. We tacitly assumed that the signal is periodic with period of T, but we would like to send different signal in the next period. If we sent different signals in every span of T, the channel could spread them over to the next span thus causing ISI. To prevent that, we use a guard interval in the time domain as before, but rather than sending a zero signal in the interval, we extend the signal with a *cyclic prefix*. If the length of the cyclic prefix d is longer than the impulse response, then the output signal sampled in the time span $[d, T + d]$ looks as if the signal was infinitely periodic. This is the principle behind multi-tone systems, in particular *orthogonal frequency division multiplexing* (OFDM), which we will focus on throughout this section. Although technically carrying multiple channel symbols, one span of T is referred to as the *OFDM symbol* the component frequencies as the *subcarriers* or *bins*. Observe that, although OFDM prevents ISI, equalization is still necessary to compensate for a non-flat frequency response which will cause different subcarriers to experience different attenuation and phase change.

Digital Modulation: In general, *modulation* refers to modifying the properties of a high frequency *carrier* signal by a lower frequency information signal. Modulation facilitates frequency division multiplexing (FDM), i.e., division of the EM spectrum into separate channels, and is used for virtually all RF communications. However, modulation can be applied within the information signal as well. Specifically, the amplitude and phase of each subcarrier in an OFDM symbol can be modulated. Equivalently, each subcarrier is represented by a pair of cosine and sine of the particular frequency, and the amplitude of each can be modulated. This is the *quadrature amplitude modulation* (QAM), which naturally fits within the OFDM scheme. By convention, a pair of real values modulating one subcarrier is represented as one complex number, or a phasor. Thus an OFDM symbol comprised of N subcarriers encodes N complex numbers. In ideal OFDM (assuming stationary channel response, perfect channel estimation, sufficient cyclic prefix), each of the $2N$ dimensions is or-

[3]These frequencies form the Fourier series.

thogonal and can be modeled as AWGN, where the noise comes from far interference and receiver circuitry. For simplicity, we assume that each subcarrier experiences equal amount of attenuation, equal amount of noise and is subject to the same power constraint.[4]

Using this model, the capacity of this channel is $C = N\log_2(1 + SNR)$ bits per symbol, but so far, the wireless channel accepts discrete-time, continuous-value inputs. To fit in the source-channel system, we need a channel coder to map sequences of bits to one or more channel symbols. In principle, this task is the dual complement of quantization in source coding. This is particularly true in the case when we contrapose a Gaussian source and a channel with Gaussian noise. In principle, the channel code defines a finite alphabet of codewords which are vectors in the multi-dimensional space R^n (or $C^{n/2}$) for block-length n. The encoder maps each sequence of bits to its corresponding codeword, e.g., an encoder taking k bits would need 2^k codewords in the codebook and achieve rate of k/n bits per channel dimension. The decoder on the other hand receives distorted codewords and needs to determine the original bit-sequence. The optimal decoder finds the *maximum likelihood* (ML) bit-sequence, i.e., the one that maximizes the conditional probability $p(\texttt{data}|\texttt{received codeword})$ When the noise is Gaussian and white (i.e., each dimension has identical independent distribution), it can be shown that the the ML decoder finds the codeword has the minimum mean square error (MMSE) from the received distorted vector, i.e., within the smallest Euclidean distance in R^n. Consequently, codewords in a good codebook are spaced as far as possible in R^n to minimize probability of decoding error, but at the same time the average power $E[X^2]$ of the codebook must be within the constraint on total power. In fact, the capacity of the AWGN channel is derived by considering the task of *sphere-packing* the R^n space, as n goes to infinity.

As in the case of compression, the scalar (one-dimensional) codebook does not achieve capacity at any SNR. Although multi-dimensional codebooks for AWGN have been proposed, e.g., *lattice codes*, the prevalent design uses two-dimensional

[4]Although that is usually not true and can be exploited as diversity discussed later, it is often the assumed model when the attenuation or interference is unknown ahead of transmission.

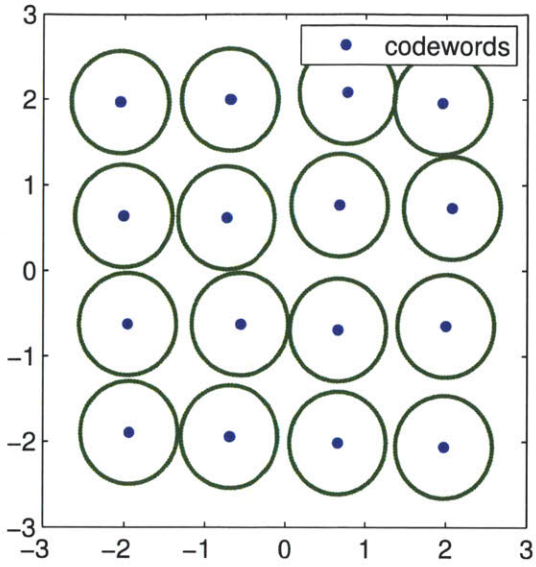


Figure 2-5: **Sphere packing in R^2.** Subject to white Gaussian noise, the error probability for any codeword is dominated by the probability of confusion with the nearest codeword. Hence, the "goodness" of the codebook is determined by the maximum radius of balls inflated around all codewords. However, the codebook is also constrained by power, the average square-distance of codewords from the origin.

codebooks which facilitate decoding via simple search. In the context of QAM, the codebook is refered to as *constellation*. Common constellations are BPSK, QPSK, 16-QAM, 64-QAM, illustrated on Fig. 2-6. Finite-block-length constellations can never guarantee correct decoding (thanks to the unbounded nature of the normal distribution), but we can compute the probability of bit error, P_b. For instance, for BPSK the probability of bit error is:

$$P_b = \frac{1}{2}\text{erfc}\left(\sqrt{SNR/2}\right)$$

where erfc is the complementary error function. One would typically define a target bit error rate, e.g., 10^{-7} and find the minimum SNR required to achieve it. Observe that the higher *order* of a constellation, i.e. the more bits are encoded in each symbol, the more closely packed the codewords need to be, thus requiring a lower noise variance (higher SNR) to achieve the same probability of bit error.

Two-dimensional constellations, limited by their short block-length, are far from achieving capacity on their own. For instance, to achieve 10^{-7} bit-error rate (BER),

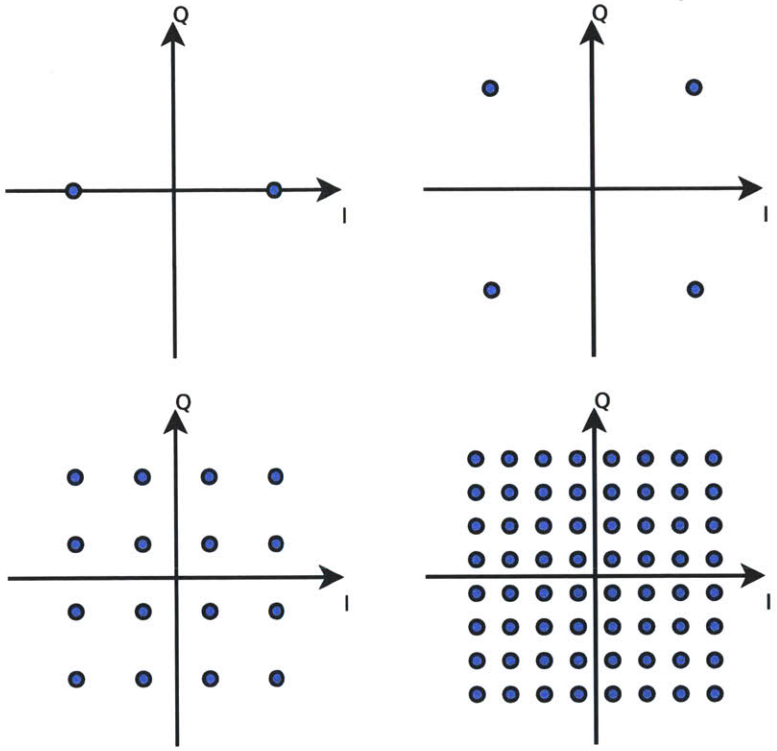


Figure 2-6: **Quadrature amplitude modulation.** In QAM, the channel codewords are fixed phase and amplitude modulations of a single-frequency carrier (e.g., one subcarrier in an OFDM symbol). Using the phasor representation of the sine wave, the codewords are complex numbers and are often drawn on the Argand plane as shown in the figure. The real and imaginary parts of the complex vector are commonly referred to as the in-phase (I) and quadrature (Q) components of the signal. The figure shows common QAM constellations: (1) BPSK, (2) QPSK, (3) 16-QAM, (4) 64-QAM. The corresponding rate is 1, 2, 4, and 6 bits per complex dimension (i.e., per Hz). The higher the rate the closer together thus less resilient to noise the codewords are, since the average power of the constellation is fixed.

BPSK requires over 10.5dB SNR to provide a rate of 0.5 bits per channel dimension. At that SNR, Shannon capacity of the AWGN channel is over 1.8 b/d. In a different comparison, Shannon capacity is 0.5 b/d at 0dB SNR, so BPSK needs 10 times more power. Rather than expecting such a low BER from the constellation we could treat the residual bit errors as a binary symmetric channel (BSC) and apply a binary code to deal with errors. The binary code could then effectively increase the block-length of the channel code, while keeping the constellation simple to implement. In practice, using weak QAM constellations to convert an AWGN channel into a BSC is not very efficient and better performance can be achieved by using soft-value decoding discussed later.

Digital Error Detection and Correction: The principle of binary channel coding is analogous to digital modulation except that, rather than in R^n, codewords are points in the n-dimensional vector space over the binary finite field where n is the block-length. Furthermore, in this space of size 2^n, there is no power constraint. In a BSC, bit-errors are iid with Bernoulli distribution, so the maximum-likelihood decoded codeword is the one from the codebook which minimizes the *Hamming distance* to the received distorted codeword.[5] Consequently, a good codebook spaces its 2^k codewords as far as possible in the 2^n space, achieving rate k/n.

Performance of binary codes is considered for a specific bit error probability on the channel, and in contrast to AWGN, one can design codes that guarantee error-free decoding when the number of bit errors introduced by the channel in a codeword does not exceed t, the error-correcting ability of the code. Minimum-distance decoding implies that to guarantee error-free output, the minimum Hamming distance between any two codewords in the codebook, d must equal at least $2t + 1$. Furthermore, the decoder can determine whether the codeword contains bit errors provided their number is less than d. For instance, a code with minimum distance 3, can detect up to 2 bit errors and correct up to 1 bit error in a block. When digital codes are used for error correction, they are often referred to as *Forward Error Correction* (FEC)

[5]Hamming distance between two binary sequences of equal length is the number of differing bits.

codes.

In 1960, Hamming formulated the *sphere-packing bound* which relates n, k and t for any binary block code and shown that there exist perfect codes that achieve this bound. For binary codes, the bound is:

$$\sum_{i=0}^{t} \binom{n}{i} \leq 2^{n-k}$$

Unfortunately, perfect codes are rare, and fortunately in practice unnecessary in the context of communications. Instead, practical error-correcting codes are designed to greatly reduce the probability of bit error while achieving a high rate.

The state-of-the-art binary codes used to strengthen the weak QAM constellation are convolutional codes and low-density parity-check (LDPC) codes [28,62]. LDPC codes are block codes which compute parity checks by adding together (modulo 2) a small number of bits from the block. Unlike block codes, convolutional codes produce a stream of parity checks computed over a short window of past data bits[6] Since binary codes use a much larger codebook than the QAM constellation, the challenge is the complexity of the decoder, as exhaustive search for the maximum likelihood codeword becomes prohibitive. To find the most likely sequence of data bits given the sequence of received coded bits, the convolutional decoder employs the Viterbi algorithm, a dynamic programming method which treats the data bits as hidden states in a hidden Markov model [97]. Unfortunately, the Viterbi algorithm requires $O(2^k)$ space for a binary code with constraint length k which limits the strength of practical convolutional codes. For LDPC, iterative belief propagation decoding methods are not optimal, but require only linear space with block-length, and perform very close to the theoretical Shannon capacity [63].

Digital codes need not be binary, but instead use a larger *finite field* as the alphabet for their arithmetic. Galois fields are of particular interest as their elements can be expressed in a round number of bits. A different bound discovered by Singleton states that for any digital code, the minimum Hamming distance d in the codebook

[6]The length of that window or "memory" is the *constraint length* of the code.

must be smaller than $n-k$, the number of parity symbols:

$$d \leq n-k-1$$

The only binary codes that are optimal under the Singleton bound (also known as *maximum distance separable* or MDS) are trivial (a single codeword or the complete field in the codebook), but there are non-trivial MDS codes for larger fields. Notably, the Reed-Solomon codes are q-ary MDS codes such that $q = 2^i$ and block-length $n = q-1$. Because Reed-Solomon codes are MDS they can correct up to $\lfloor (n-k)/2 \rfloor$ symbol errors where k is the number of data symbols per block, i.e. at rate k/n. However, observe that as each symbol is composed of i bits, an error in any of the bits constitutes a symbol error. Reed-Solomon codes are typically decoded in $O((n-k)^2)$ time using the algebraic Berlekamp-Massey algorithm.

Code Concatenation and Interleaving: Although we suggested that the residual error after decoding QAM can be treated as BSC, in a higher order QAM multiple bits are encoded per symbol, and hence one symbol error can map to multiple bit errors. Using Gray coding for mapping bits to QAM symbols ensures that if the decoder mistakes the symbol for an immediately adjacent symbol then exactly one bit in the symbol is incorrect. However, if the distorted symbol lies farther away, we get potentially a burst of bit errors. Furthermore, as we will discuss in the next section, *fading* may lead to burst errors. A burst of errors within a block or constraint length of the FEC code will likely exceed the code's error correcting ability, hence the goal is to distribute the errors over multiple blocks. This channel decorrelation is dual to source decorrelation described in Section 2.1.2, and aims at converting the non-memoryless channel into a memoryless BSC. The common approach to decorrelation is *interleaving* which in essence permutes the symbols in the stream in deterministic or pseudo-random fashion.

This general strategy of combining two channel codes with interleaving in-between is called *code concatenation*. This technique can also be applied to correct residual

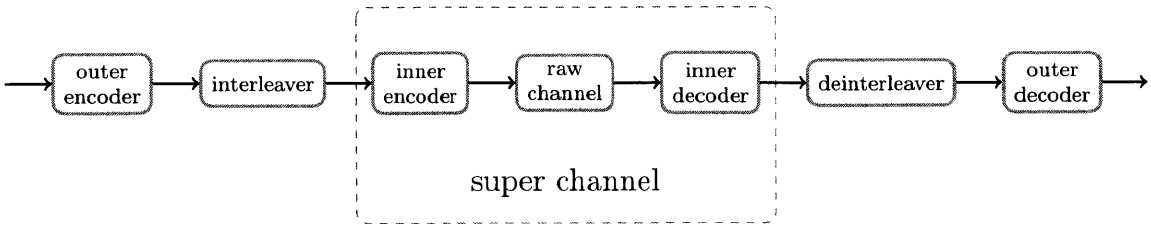


Figure 2-7: **Code concatenation.** The inner code wraps the raw channel into a "super-channel" which exhibits lower noise or error probability but also lower rate. The outer code encodes for the super-channel to remove any residual errors. to spread any bursts in the residual errors, an interleaver is applied in-between. At the encoder the outer code is applied first, followed by the interleaving, and finally inner code. The process is reversed at the decoder.

errors after FEC. When FEC decoder fails to decode all errors, there are typically a large number of errors in a block or a burst of errors in a convolutional stream. Although the rate of the resulting code is the multiple of rates of its components, thus smaller, the effective block-length is large without paying the price in complexity as shown by Forney. The disadvantage, however, is the increased latency needed to interleave and deinterleave the blocks.

Soft-Decision and Iterated Decoding: Consider the concatenation of QAM with a convolutional FEC. For practical reasons, the early decoders decoded the QAM symbols to bits and then the resulting bit-stream into data bits. In this *hard-decision* system the received symbol in R^n is converted to a codeword in 2^k then passed to the FEC decoder and all other information about the original symbol is forfeit. Therefore, the FEC decoder sees a BSC and each coded bit is considered to carry the same amount of information. In particular, the FEC decoder cannot distinguish whether the received QAM symbol is near or far from a constellation point. However, if the received QAM symbol is far from its presumed original, then the conditional probability that this is the actual transmitted symbol is low. Consequently, the confidence in the decoded symbol is low. Given this confidence information, an FEC decoder can perform significantly better. In particular, recall that a block code which can detect $d-1$ errors, can correct $\lfloor (d-1)/2 \rfloor$ errors. Furthermore, when applied to a binary *erasure* channel (BEC), the code can recover from $d-1$ erasures. If

the decoder is told which bits have low confidence, it can treat them with erasures and correct them with much greater effectiveness than unknown bit errors. Thus, modern FEC codes use *soft-decision* decoding in which the information exchanged between the inner and outer code is composed of *soft values* indicating the inner decoder's confidence. Intuitively, a soft-decision concatenated code is an efficient implementation of a vector modulation. A more general approach is to jointly design the FEC codebook and modulation constellation. An example of such design is *trellis-coded modulation* (TCM) in which the constellation symbols directly encode parity checks on the data stream [93]. In this case, the joint decoder accepts as input the received constellation symbols and uses these observations to determine the most likely sequence of data bits. Soft-decision decoding can also be part of an iterated decoding algorithm such as belief propagation in LDPC codes or the turbo method in turbo codes [8,63]. Turbo codes involve random interleaving of concatenated encoders at the transmitter and iterative maximum a-posteriori (MAP) decoding [8].

Errors as Erasures: No matter how much redundancy is introduced to the channel symbols, when the channel capacity is lower than effective rate, Shannon's theorem shows that errors in the decoded data stream are inevitable. In most cases, any corruption to the data is catastrophic, whether the flipped bit is a control flag, part of an integer in a data structure, a floating point value or a codeword in arithmetic or variable length code. Thus, to protect from such failures, an error detecting code is used. Packets which fail the parity check are then discarded. Thus, a BSC can be converted into a BEC. This conversion carries a trade-off: all data bits covered by share fate with just a single bit error.

In comparison to FEC, the rate of the error detecting code can be very high. Furthermore, the code is typically fine tuned to the expected error patterns. For example, the popular 32-bit CRC-32 is used on packets of around 12000 bits in length and is designed to detect bursts of errors.

Capacity Estimation and Adaptation: For efficient communication in the separate design, the channel coder must adjust the code rate to the capacity of the channel. If the rate is too low, the communication is not as efficient as it could be. On the other hand, if the rate is too high, the error probability is significant and data corruption is inevitable. There are essentially two ways of dealing with variable capacity.

Rate adaptation: the receiver assesses the state of the channel and communicates it to the transmitter which adjusts the channel code rate. Hints that can be used for capacity estimation include estimated SNR (e.g., from training sequences and pilots), BER or single-bit packet erasure state (i.e., reception acknowledgment or ACK). One of the main challenges of rate adaptation in wireless networks is to separate effects that can be mitigated without modifying the channel code, such as interference from other users of the medium in TDMA or CSMA schemes.

Rateless coding: the encoder uses a scheme in which the codeword carrying given message can be extended on demand. Thus the effective rate is progressively reduced, until the receiver successfully decodes the message and acknowledges to the transmitter that no more symbols are needed. Notably, most wireless systems employ ARQ (Automatic Repeat Request) which is the simplest repetition-based rateless scheme. Rateless codes for erasure channels have received a lot of attention in the context of packet networks. For BEC, hybrid techniques such as HybridARQ are the mid-point between rateless coding and fixed-rate coding. Recently, the rateless approach has been applied to real-valued noise channels, e.g. AWGN [22,82].

2.2.3 Fading

As discussed in the previous section, the received RF signal is a superposition of multiple copies propagating along different paths through the environment. This leads to ISI and frequency-selective attenuation which requires equalization. This effect also applies to other media, such as cables, as well. However, in the wireless setting the attenuation along any path is inherently variable as the device antennae or parts of the environment are mobile. The result is that the channel exhibits *fading*,

i.e., significant deviations of the effective attenuation.

Whether caused by multi-path interference or line-of-sight obstruction (*shadowing*), fading is inherently connected to the phase-geometry of the environment. Thus, fading can be observed as we move along different dimensions: time, frequency and space.

How fading is addressed depends on the speed of the variation. If the changes are sufficiently slow to estimate the new value, communicate it via feedback and respond by adjusting the channel coding scheme for the next coding block, then they can be dealt via adaptation in a feedback loop. Rate adaptation, for instance, addresses slow fading. Similarly, given frequency-selective slow fading, the OFDM channel encoder could allocate different code rates to different frequency bins.

On the other hand, if the fading is fast then the feedback loop is too slow to adjust code rate in response to capacity changes. There is high probability that the channel will experience a *deep fade* during a code block which will lead to data corruption in that block, i.e., *outage*. To prevent outages, the channel encoder must ensure that each coding block experiences average channel state. Thus, the coder uses an interleaver or another decorrelation method to spread each deep fade across many blocks. This way the variance of the channel conditions across blocks, and thus the probability of outage is significantly reduced.

Observe that although interleaving increases latency of the code, it does not increase the block-length. In particular, interleaving applied to a memoryless channel does not change the capacity or the performance of any code. However, if the underlying channel exhibits fading, then the *outage capacity* is improved by interleaving. The same principle is exploited in the code concatenation technique discussed earlier. In fact, if interleaving of the channel symbols is not feasible, the channel coder can deal with block failures of the channel code by interleaving its input and applying an *outer code* as in concatenation. This way errors from the failed block will be distributed across many blocks of the outer code and recovered from.

2.2.4 Coded OFDM-based Communication System

In this section, we put together the techniques introduced above to describe the design of a typical modern wireless communication system. We focus our attention on the popular OFDM-based design common to the PHY layer of 802.11, 802.16 and DVB wireless standards. In these standard, the communication takes place in the UHF and SHF radio frequency ranges, from 500MHz to 5GHz, but is confined in channels 5MHz to 40MHz wide. Therefore, the typical implementation uses analog signal processing to upconvert/downconvert the baseband signal to the communication passband, while all baseband processing is digital, requiring digital clock speeds below 100MHz. The scope of this exposition is the digital signal processing where virtually all of the channel coding is applied.

Figure 2-8 presents a conceptual block diagram of the coded OFDM PHY layer. The transmitter converts packets of bits to frames of digital baseband samples. The receiver processes the stream of digital samples to detect frames and decode packets to be passed to the higher layers. The processing blocks are organized into a pipeline with notable correspondence between the encoding and decoding blocks of specific features of th channel code. We follow the feature stack from the raw baseband channel at the bottom to the packet erasure channel presented to the higher layer. This order matches the layout of Section 2.2.2 for the convenience of the reader.

Preambles and Training: To help the receiver identify the beginning of a frame, i.e., wanted signal, from the stream of baseband samples, the transmitter prepends a *preamble* of known symbols to the frame. For simplicity of logic design these symbols are often unique so that the detection is not triggered in the middle of a frame. The key property of the preamble is that it can be detected despite unknown channel response because the distinguishing characteristic survives any linear time-invariant (LTI) channel. Such preamble also allows timing recovery, i.e., establishing symbol boundaries. For example, the 802.11a/g preamble is composed of so called "short training symbols" which are periodic with half-symbol period. Since this periodicity is preserved by the channel, the receiver can detect such symbols by observing spikes in

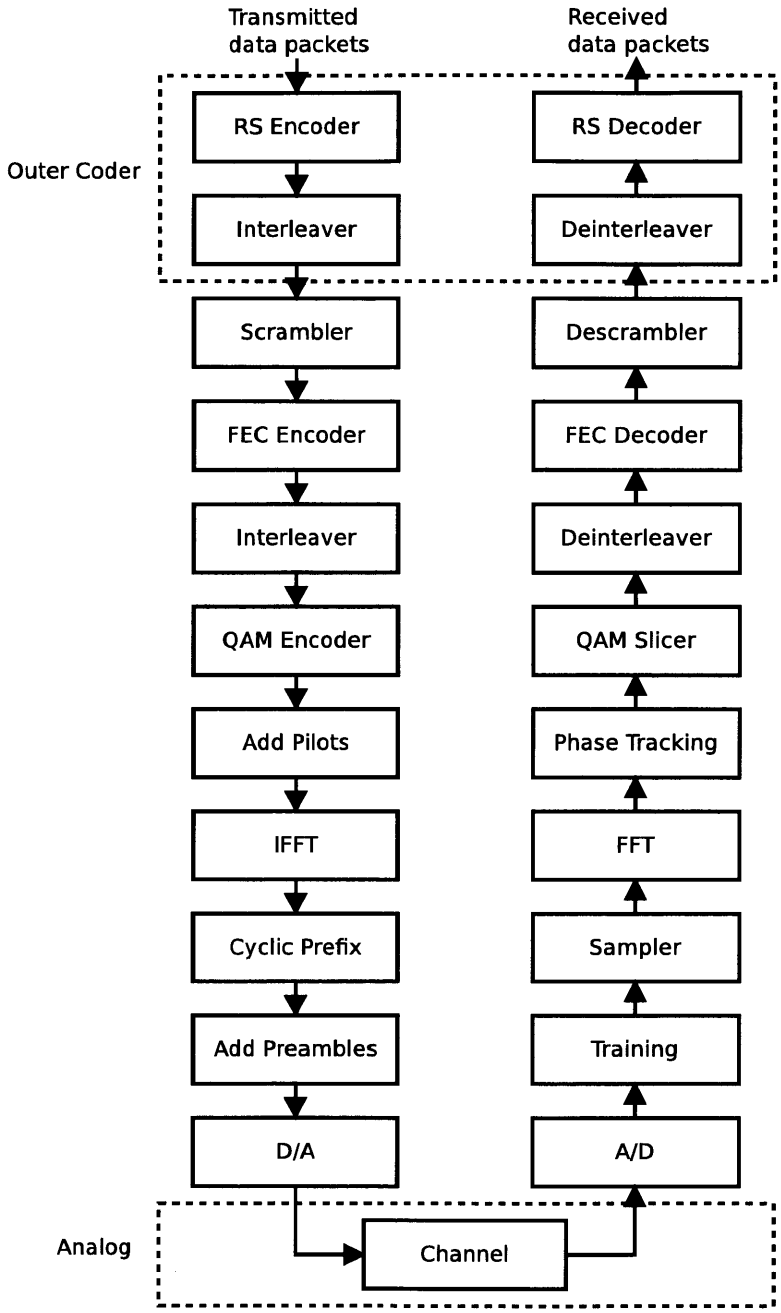


Figure 2-8: **Organization of an OFDM-based channel coder.** E.g., 802.11a/g/n (without the outer code) or DVB-T. Retransmissions and bit-rate adaptation are performed at the higher layers: link and MAC.

the auto-correlation function in the signal, e.g., as in the Schmidl-Cox synchronization algorithm.

The preamble can also be used to determine the channel response and train the equalizer. In an OFDM system, this is commonly done in a coherent fashion in the frequency domain, i.e., after synchronization, sampling and FFT. The receiver simply compares the received symbol to the expected known one to determine the linear, but frequency-dependent channel response which can be described by one complex number per frequency bin.

Finally, a practical wireless channel has non-LTI effects such as a carrier frequency offset (CFO) or a sampling frequency offset (SFO). Such offset is caused by a mismatch in the hardware oscillators between devices or the Doppler effect, and results in a frequency shift or time-dependent phase offset. Both offsets can be estimated by observing the channel impact on the known preamble symbols.

Cyclic Prefix and Sampling: The cyclic prefix is the ISI-free guard interval (GI) between OFDM symbols. In 802.11a/g, the cyclic prefix adds 1/4th to the symbol length. Specifically, the 20MHz channel is sampled at 40Msamples/s and one OFDM symbol spans $4\mu s$, i.e. 80 samples. Of those, 16 samples are the cyclic prefix in front of a 64-sample data symbol. In 802.11n and DVB-T other choices for the GI are available thus providing a tradeoff between delay-spread tolerance and efficiency. At the receiver, the signal from the timing recovery indicates which subsequences of samples should be considered as OFDM symbols. Recall that to guarantee that to remove the effects of an channel with impulse response of length d, the sample needs to be done at least d after the transmission of the previous symbol has ended. If d is shorter than GI, the sampler has some slack.

IFFT and FFT: The key component of the OFDM pipeline is the conversion between the time domain of the baseband signal and the frequency domain to which the data is mapped. The transmitter converts the frequency bins to time-domain signal by applying complex inverse digital Fourier transform (IDFT). The receiver

simply inverts the operation by applying complex DFT. Fast Fourier transform (FFT) is the common algorithm for computing these transforms. In 802.11, the 64 samples in an OFDM symbol are converted to 64 frequency bins. Observe that since the sampling frequency is 40MHz, the subcarriers are $\frac{40/2}{64} = 312.5$kHz apart.

Pilots and Phase Tracking: Not all frequency bins are used for data. Specifically, some bins are left empty for spectrum shaping, i.e., ensuring that the signal fits the spectral envelope imposed by the RF spectrum regulator (such as the FCC) and that there is no aliasing in case of severe CFO. In addition, some bins are reserved for *pilot tones*. Those known values in specific bins can be used by the receiver to track changes in the channel response from one OFDM symbol to the next. For instance, a residual CFO causes the phase offset of all bins to advance between symbols, while a residual SFO causes a linear phase offset. 802.11 uses only 52 out of the 64 bins and reserves 4 bins for pilots, thus leaving 48 bins for data. Together with GI, bin allocation in 802.11 reduces the net data bandwidth of the 20MHz channel from nominal 20M to 12M complex dimensions per second.

QAM Mapper and Slicer: The QAM modulation is where bits are mapped to the waveform in the OFDM bins. The most popular constellations, employed both in 802.11 and DVB systems are shown in Fig. 2-6. Observe that higher order constellations carry more bits per dimension, but also require more bits of precision to express the waveform value. In practice, a fixed number of bits is used for the waveform (both before and after FFT) and is often tuned to the receiver noise floor. The receiver performs demapping or slicing to recover the original bit-stream by finding the nearest (in Euclidean distance) constellation point to the received value. In most modern implementations, the slicer outputs a *soft-value* per bit rather than 0 or 1. This allows soft-decision decoding which greatly reduces BER. Notably, in 802.11 the order of the QAM as well as the rate of the FEC code described below can vary from packet to packet. Therefore, the packet must include a SIGNAL field which is always mapped using the same QAM constellation and FEC code. Parameters that don't

change from packet to packet are typically announced via periodic beacons from the wireless access point or base station.

FEC Code: 802.11a/g and DVB-T employ a binary convolutional code (BCC) with rate 1/2 and constraint length 8. For greater flexibility, the code can be *punctured* by removing some of the check bits from the code stream. This increases the data rate at the price of reducing minimum distance between codewords thus increasing BER or conversely increasing the required SNR to achieve the same BER. More advanced 802.11n, 802.16 and DVB-T2 employ LDPC or Turbo codes.

Interleaving: As the frequency response of the 20MHz channel is rarely flat, the encoder must ensure that bits within the constraint length of the BCC do not experience a deep fade thus causing a bit error in the decoded data stream. This is achieved by interleaving the coded bits across OFDM bins before they are mapped via QAM. Additionally, not all bits in a higher order Gray-coded QAM are equally protected, and therefore the interleaver ensures that highly protected bits are interspersed with more noisy bits in order to average the behavior. Notably, the LDPC code has codeword block-lengths starting at 648 bits which is greater than 192, the maximum number of coded bits per OFDM symbol at the highest order 64-QAM, and therefore is not interleaved.

Scrambler: To reduce the probability of long sequences of zeros or ones in the input of the FEC coder, the scrambler decorrelates the input sequence by multiplying the input signal with pseudo-random binary noise sequence.

Outer Code: In some systems, such as DVB-T and DVB-T2, but also in some 802.11n modes, an outer coder is applied to reduce the probability of residual bit errors. Reed-Solomon code can be applied on top of BCC with or without interleaving. Interleaving is not required to yield benefits from code concatenation, because the errors in the output of the BCC are most likely bursty and the Reed-Solomon code (RSC) operates on byte sized symbols. Therefore, 8 bit errors in a row will corrupt

at most two symbols in a RSC block and can be easily corrected. In DVB-T, the BCC and RSC are interleaved using a convolutional interleaver which spreads the RSC encoded bytes from one 204-byte block as far as 2244 bytes apart.

Error Detection: To guarantee that the packets delivered to the higher layer are error free, a CRC32 check is appended to the data frame. If the check fails, the received frame is discarded and no ACK is sent. Lack of ACK is interpreted by the transmitter as reception failure and the frame is retransmitted. This simple feedback is the basis of bit-rate adaptation in 802.11a/g. In 802.11n, the communicating devices can exchange souding frames thus providing more accurate estimate of channel state including per-bin information.

2.3 Joint Source-Channel Coding

The separation theorem, based on ideas from Shannon [86], proves that the designs of the source and channel codes can be decoupled without loss of optimality if the channel capacity is well defined, as in the case of stationary point-to-point channels. Such separation greatly simplifies the construction of the system by focusing the concerns of the code designers on only one source of uncertainty, either the source or the channel. However, such approach also has some drawbacks. In particular, channel coding assumes that the channel statistics are known to the source. When this assumption is violated, i.e., the actual channel capacity drops below target capacity, the performance of these codes degrades sharply, showing a *threshold effect.* This threshold degradation is a direct result from the separation of channel and source coding, where errors at the channel decoder may render the output of the source decoder completely random. The second drawback is that the separation principle applies only to stationary point-to-point channels and could incur a severe penalty for other channel types [30]. Last, digital communication codes usually requires long blocklengths and high encoder/decoder complexity to achieve near-optimal distortion which becomes problematic for real-time communication.

The above limitations of digital separation are particularly important for wireless channels. Specifically, in mobile scenarios the channel SNR may vary widely and unpredictably [98], creating a mismatch between the instantaneous channel capacity and the target capacity of the used channel code. To prevent such a situation, a digital design may need to be overly pessimistic in its choice of channel-code rate. Broadcast channels are also problematic for digital design because receivers may have diverse channel SNRs, invalidating the assumptions of the separation theorem. Finally, in sensor networks and mobile applications, latency and computational complexity become important design decisions. As a result, one may prefer a mild increase in distortion if it results in a reduction in latency or computational complexity. To address the deficiencies of the digital separation, we have to jointly optimize the source and channel codes and the result falls into *joint source-channel coding* (JSCC). Technically, any joint or cross-layer design and optimization of source and channel coders falls into JSCC making it a rather broad term. In this chapter, however, I will focus on the techniques addressing the case of transmitting analog signals (such as video) on broadcast channels which are most relevant to this thesis.

2.3.1 Multi-Resolution (Layered) Coding

When users with different channel capacities want to receive data from a single source, the user with higher capacity expects a more accurate (less distorted) content, but otherwise there is a high correlation between the reconstructions that each user receives. Thus, if the system is to support such users simultaneously and the channel resources are limited, the natural approach is to attempt to isolate common information that can be delivered to all users to avoid redundant transmissions. On top of that, the better user using the extra capacity receives additional data which allows it to reduce the distortion in reconstruction.

For this approach, commonly referred to as layered or multi-resolution coding, to be feasible, the channel must allow a channel signal to be received by all users, which in general is not always the case. However, many broadcast channels are *degraded* [16], that is, the different realizations of the channel can be modelled as simply successively

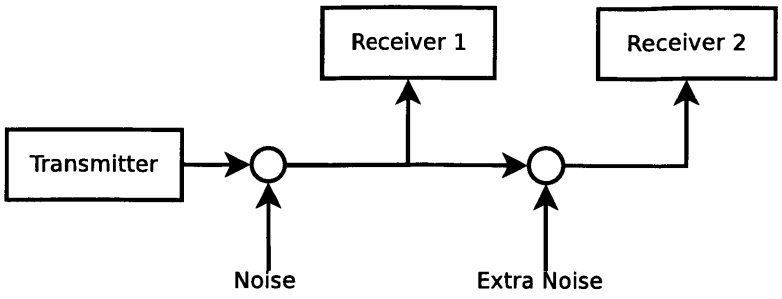


Figure 2-9: **Degraded broadcast channels.** In a degraded broadcast channel, the user with a weak channel (Receiver 2) receives the same signal as the user with a strong channel (Receiver 1) but with additional noise. In a wired channel, the noise equivalent is packet loss.

accumulating more channel noise. This is illustrated in Fig. 2-9. A wired broadcast network can be *physically* degraded in the sense that all packets that reach the second user first go through the first user which acts as a forwarder, as in application-level multicast [12,26]. In contrast, in a wireless broadcast, users experience independent channel noise coming from nearby interferers or their receiver circuitry. However, in many cases (as in the case of the AWGN channel), such broadcast channel is *stochastically* degraded, that is, it can be modeled as in Fig. 2-9.

When a channel is degraded, the transmitter can encode the data into a base layer, targeting all users and carrying coarse approximation of the source signal, and a series of enhancement layers which progressively refine the quality of the decoded signal. The more layers a user receives, the better its reconstruction. Most importantly the layers form a stack in the sense that an enhancement layer is only useful if all the coarser layers have been received without error.

A distinguishing feature of layered coding is that it allows to keep some notion of the source-channel separation. The source coder deals with the statistics of the source and generates layers of codewords that allow progressive refinement of its reconstruction, while the channel coder deals with the statistics of the channel to ensure that the base layer is better protected from channel noise and thus reaches all users. However, in contrast to the point-to-point (single layer) scenario, both coders have to jointly optimize the bit rate of each layer since the layers essentially interfere with each other on the channel, that is any channel resources (power, bandwidth or

airtime) spent on one layer require a reduction in bit-rate of the other layer.

Successive Refinement: Recall that in the process of lossy compression described in §2.1.2, the source coder discards some information about the source signal in order to describe it using the finite number of bits in a message. The decoder then approximates the original values from the binary codewords it receives. To support layered coding, the source coder creates a coarse approximation of the data using a few bits and then iteratively refines the approximation using the additional bits. In information theory, *successive refinement* refers to the problem of finding such layered source code that the distortion achieved at any stage is optimal [21]. Formally, if the source signal has a distortion-rate function $D(R)$ and layer i is encoded at bit-rate R_i, then the distortion achieved by the user who receives the first k layers would be

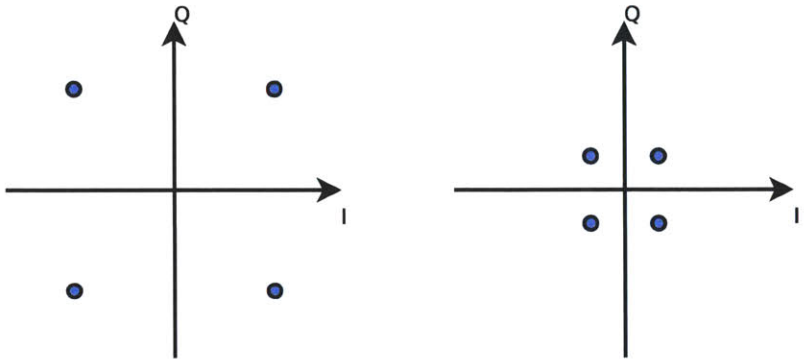
$$D = D\left(\sum_{i=1}^{k} R_i\right)$$

A small number of source distributions have been shown to be successively refinable with respect to specific distortion metrics [21,92]. In general, the key to successive refinement as in achieving the optimal rate-distortion trade-off is using long blocks rather than single source symbols. That said, a simple implementation of a layered quantizer could simply use the quantization error of the previous layer as the input signal to quantize again. Such design does not generally achieve successive refinement, but is easy to implement. For instance, the *SNR scalable* profile of the work-in-progress H.264/SVC (Scalable Video Coding) standard [85] applies this approach to the DCT coefficients after inter-frame prediction. A different method is to leverage multi-resolution properties of some transforms. For instance, the coefficients of the wavelet transform used in the JPEG2000 standard can be split into different bins ranging from coarse to fine so that reconstructing the image from a subset of bins produces a blurry approximation of the original.

Superposition Coding: Once the source prepares a base and a series of enhancement layers, the channel coder must ensure that each layer is appropriately protected from channel noise. Unlike the point-to-point case where the channel code simply aims at zero error-rate and protects all input bits equally, a layered channel coder has to provide *unequal error protection* (UEP). A simple approach to achieve different resilience to noise is to use time-division multiplexing between different channel codes. The base layer is encoded using at a low rate using a strong channel code that guarantees resilience at low channel SNR, while the enhancement layers are encoded using at a high rate using a weak channel code thus only decodable at high SNR. This approach however is rather inefficient and strictly better rates can be achieved using *superposition coding* in which the codewords of each layer are added together in the channel signal space [16].

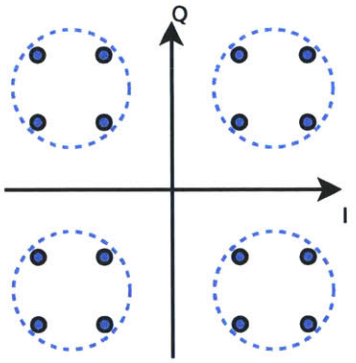
Hierarchical Modulation: The idea of superposition coding can be applied to QAM (see *Digital modulation* in §2.2.2) to create hierarchical modulation. When codewords of two constellations are added together, the result is a Cartesian product of possible codewords as illustrated in Fig. 2-10. Crucial for unequal error protection of the constellations added together is that they are scaled accordingly. The constellation encoding the high priority bits in the base layer is allocated more power by scaling it by a larger factor than the low priority enhancement constellation. This allows a low SNR receiver to differentiate between clusters corresponding to different symbols in the base constellation even if the noise corrupts the enhancement symbols to the extent that decoding is impossible.

Observe that in hierarchical modulation, the layers interfere with each other. If the receiver cannot distinguish between the codewords of the low priority constellation, then the unknown symbols directly contribute to the noise from the channel. Compare the minimum distance in the base constellation in Fig. 2-10a to the combined constellation in the case where the receiver only attempts to decode which cluster the codeword belongs to. Thus, the presence of the enhancement layer increases the minimum SNR required for the base layer to be decodable at a desired BER. We



(a) High Priority Base

(b) Low Priority Enhancement



(c) Superposition

Figure 2-10: **Hierarchical Modulation.** Two QPSK constellations are added together with different scaling factors to form the hierarchical constellation. At high SNR, the receiver can resolve all codewords, but at low SNR it can only resolve which cluster (circled) the codeword belongs to as the noise is likely to confuse the codewords within each cluster.

can reduce the interference by scaling the enhancement constellation down, but that increases the minimum SNR required to decode that layer. It is important to note that this *inter-layer interference* is present in other unequal error protection schemes, such as time-division multiplexing or embedded diversity coding [20].

2.3.2 Analog-like and Hybrid Digital-Analog Coding

When transmitting an analog signal such as video over an analog channel such us wireless, it is worth considering to directly map the source signal to the channel signal using a continuous mapping. By avoiding quantization at the transmitter, the distortion of the reconstruction is not determined by the chosen bit rate, but only by the amount of noise on the channel. This approach naturally provides graceful degradation and can simultaneously support diverse receivers.

In fact, when transmitting a scalar white Gaussian source over am AWGN channel, one can easily show that a very simple linear encoder and decoder which only scales the input signal by a factor can achieve the optimum performance [7,16,34]. In contrast to the separate design which requires unbounded block-length to achieve optimal performance at a specific known SNR, the linear design has a unit block-length and operates at optimum regardless of the actual SNR without any feedback or adaptation.

Although other combinations of sources and channels where such *uncoded* transmission is optimal have also been also explored [29], to enjoy this remarkable property the statistics of the source and channel need to correspond to each other in a so called match. Specifically, if the channel is multidimensional with AWGN, then a matching source is a multivariate iid Gaussian variable with the same number of dimensions. This is unlikely in practice, and if there is a gap in dimensionality, also known as *bandwidth mismatch*, the linear design inevitably becomes suboptimal.

As simple linear scaling cannot effectively address a bandwidth mismatch, many non-linear schemes have been proposed [15,38]. These schemes achieve bandwidth expansion or compression by using space-filling curves to map between one-dimensional and two-dimensional spaces. Alternative hybrid designs combine digital and analog (most often linear) coding to leverage the efficiency of the digital design with

some graceful degradation or improvement coming from the analog design. Generally, these schemes operate close to the optimal performance while providing some graceful degradation [77,88]. However, they cannot achieve the remarkable property of uncoded Gaussian transmission which is simultaneously optimal at all receiver SNRs.

Chapter 3

SoftCast: an End-to-End Linear Design

This chapter presents SoftCast, a clean-slate end-to-end architecture for transmitting video over wireless channels. In contrast to the conventional design which separates the concerns of the video compression codec from the error correction codes of the PHY, SoftCast adopts a unified design that both encodes the video for compression and for error protection. This end-to-end approach enables a one-size-fits-all video stream that can deliver multicast video to multiple mobile receivers, with each receiver obtaining video quality commensurate with its specific instantaneous channel quality.

3.1 Why Does the Conventional Design Not Allow One-size-fits-all Video?

Today's approach to compression and error protection coding prevents existing wireless design from providing one-size-fits-all video.

(a) Compression: Video pixels are highly correlated within a frame. Further, video frames are correlated in time. MPEG exploits this correlation by operating on sequences of successive video frames called GoPs (Group of Pictures). MPEG com-

presses a video in two steps [27]. First, it performs intra-frame compression to remove redundant information within each frame. This is done by applying a 2-dimensional DCT on small blocks of 8x8 pixels, and quantizing the resulting DCT components to a fixed precision. The conventional design then treats these quantized real values as sequences of bits and compresses them to a compact bit sequence using a Huffman code. Second, MPEG performs inter-frame compression to eliminate redundant information across frames in a GoP. In particular, it uses differential encoding, which compares a frame against a prior reference frame and only encodes the differences. It also uses motion compensation to predict the movement of a particular block across time. Using this combination MPEG achieves good compression ratios. However, it is this combination that prevents one-size-fits-all video:

- Quantization is performed by the source, and coarsens the resolution of the video to match a desired bit rate, and hence fixes the quality of the video, even if the receiver channel could support a higher bit rate.

- Huffman coding and differential encoding fail sharply in the presence of bit errors and packet losses. Specifically, a Huffman code is variable length, and a few bit flips can cause the receiver to confuse symbol boundaries, making the whole frame unrecoverable. Differential encoding and motion compensation create dependencies between different packets in a coded video, and hence the loss of some packets can prevent the decoding of correctly received video packets.

Note that layered and scalable video coding (SVC) also use quantization, variable-length coding, differential encoding and motion compensation, and hence are also highly sensitive to wireless errors.

(b) Error Protection: Error protection is typically done at the physical layer (PHY) by picking a bitrate, i.e., a combination of modulation and forward error correcting code, that ensures the receiver can decode the vast majority of the packets correctly. The packet decoding probability drops sharply when the bitrate chosen is higher than can be supported by the channel SNR [73], and hence the PHY layer is

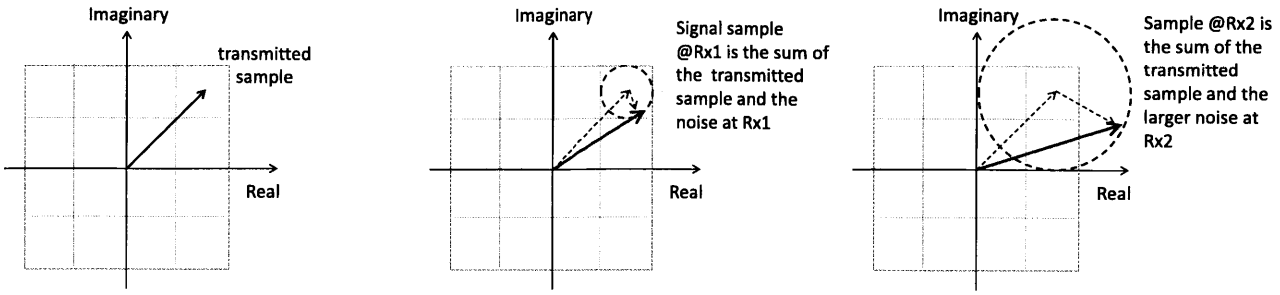
constrained to pick a low modulation and code rate that works well across time and receivers.

The problem occurs when the actual channel conditions are too bad for the bit rate that the PHY chose, and today, there are two popular ways to approach this problem:

Bit rate adaptation in which the PHY attempts to predict which bit-rate will yield the maximum throughput. In response, the application must monitor (often indirectly, e.g., by observing buffer levels) the effective transfer rate and adjust the level of source compression when necessary, in order to ensure timely streaming. In effect, the end-to-end delay inherently includes the delay in this feedback loop. This approach also requires that the adaptive stream is dedicated to one user which is unscalable.

Layered coding in which the PHY anticipates multiple simultaneous receivers with different channel conditions, and therefore cannot pick a single bit-rate for all. Instead, the application encodes the source content into ordered layers, so that the more layers delivered, the better the quality of the decoded content. Each layer is then encoded by the PHY in such way that the first (base) layer should be decodable without errors by all receivers, but decoding the further (enhancement) layers would require better channel quality. And so, the better the channel conditions at a particular receiver, the more layers can be decoded, yielding a higher quality of the content delivered.

Although layered coding, e.g., hierarchical modulation [56], in combination with a layered video codec [85] can provide some graceful degradation, such system is inherently inefficient. The core of the inefficiency lies in that which layers can actually be decoded is determined by the channel quality at the receiver, after the wireless resources (power, bandwidth and airtime) have already been spent. This is in contrast to layered multicast problem in wired networks where layers beyond the available capacity can be simply dropped from the packet queue without causing any congestion to higher priority base layer [3]. On a wireless channel, each layer consumes some of



(a) Transmitter (b) Nearby Receiver (c) Far Receiver

Figure 3-1: **Wireless broadcast delivers more signal bits to low noise receivers.** The figure shows the transmitted sample in red, the received samples in blue, and noise in black. The source transmits the signal sample in (a). A nearby receiver experiences less noise and can estimate the transmitted sample up to the small square, i.e., up to 4 bits. A far receiver sees more noise and hence knows only the quadrant of the transmitted sample, i.e., it knows only 2 bits of the transmitted sample.

the limited resources and so depending on the resource allocation, either users with good channels or those with weak channels will be favored. This also means that the PHY cannot optimally select layer bit rates without coordination with the application layer.

3.2 SoftCast Overview

SoftCast's design harnesses the intrinsic characteristics of both wireless broadcast and video. The wireless physical layer (PHY) transmits complex numbers that represent modulated signal samples, as shown in Fig. 3-1(a). Because of the broadcast nature of the wireless medium, multiple receivers hear the transmitted signal samples, but with different noise levels. For example, in Fig. 3-1, the receiver with low noise can distinguish which of the 16 small squares the original sample belongs to, and hence can correctly decode the 4 most significant bits of the transmitted sample. The receiver with higher noise can distinguish only the quadrant of the transmitted signal sample, and hence can decode only the two most significant bits of the transmitted sample. Thus, wireless broadcast naturally delivers to each receiver a number of signal bits that match its SNR.

Video is watchable at different qualities. Further, a video codec encodes video at different qualities by changing the quantization level [27], that is by discarding the least significant bits. Thus, to scale video quality with the wireless channel's quality, all we need to do is to map the least significant bits in the video to the least significant bits in the transmitted samples. Hence, SoftCast's design is based on a simple principle: ensure that the transmitted signal samples are linearly related to the original pixel values.

The above principle cannot be achieved within the conventional wireless design. In the conventional design, the video codec and the PHY are oblivious to each other. The codec maps real-value video pixels to bit sequences, which lack the numerical properties of the original pixels. The PHY maps these bits back to pairs of real values, i.e., complex samples, which have no numerical relation to the original pixel values. As a result, small channel errors, e.g., errors in the least significant bit of the signal sample, can cause large deviations in the pixel values.

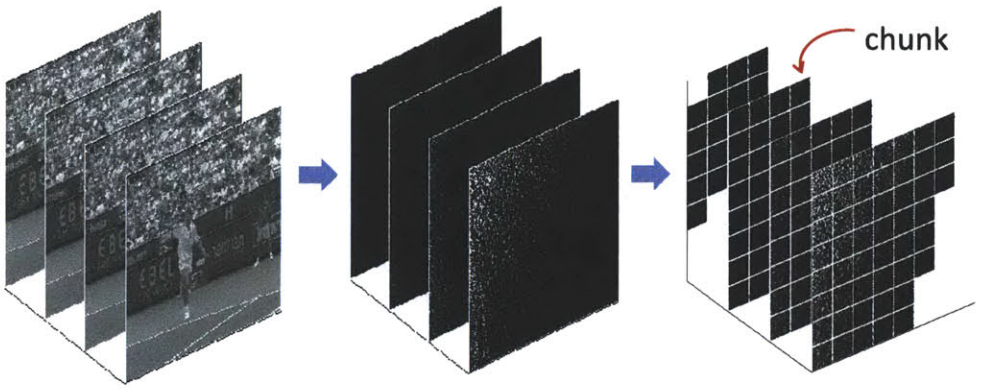
In contrast, SoftCast introduces a clean-slate joint video-PHY architecture. SoftCast both compresses the video, like a video codec would do, and encodes the signal to protect it from channel errors and packet loss, like a PHY layer would do. The key characteristic of the SoftCast encoder is that it uses only linear codes for both compression and error and loss protection. This ensures that the final coded samples are linearly related to the original pixels. The output of the encoder is then delivered to the driver over a special socket to be transmitted directly over OFDM.

3.3 Encoder

SoftCast's encoder both compresses the video and encodes it for error and loss protection.

3.3.1 Video Compression

Both MPEG and SoftCast exploit spatial and temporal correlation in a GoP to compact information. Unlike MPEG, however, SoftCast takes a unified approach to intra



(a) 4-frame GoP (b) 3D DCT of GoP (c) Discarding Zero-Valued Chunks

Figure 3-2: **3D DCT of a 4-frame GoP.** The figure shows (a) a 4-frame GoP, (b) its 3D DCT, where each plane has a constant temporal frequency, and the values within a plane represent spatial frequencies at that temporal frequency, (c) the non-zero DCT components in each plane grouped into chunks. The figure shows that most DCT components are zero (black dots) and hence can be discarded. Further, the non-zero DCT components are clustered together.

and inter-frame compression, i.e., it uses the same method to compress information across space and time. Specifically, SoftCast treats the pixel values in a GoP as a 3-dimensional matrix. It takes a 3-dimensional DCT transform of this matrix. The DCT transforms the data to its frequency representation. Since frames are correlated, their frequency representation is highly compact.

Fig. 3-2 shows a GoP of 4 frames, before and after taking a 3D DCT transform. The grey level after the 3D DCT reflects the magnitude of the DCT component in that location. The figure shows two important properties of 3D DCT:

(1) Most DCT components have a zero (black) value, i.e., have no information. This is because frames tend to be smooth [99], and hence the high spatial frequencies tend to be zero. Further, most of the structure in a video stays constant across multiple frames [27], and hence most of the higher temporal frequencies tend to be zero. This means that one can discard all of these zero-valued DCT components without affecting the quality of the video.

(2) Non-zero DCT components are spatially clustered. This is because spatially nearby DCT components represent nearby spatial frequencies, and natural im-

ages exhibit smooth variation across spatial frequencies. This means that one can express the locations of the retained DCT components with little information by referring to clusters of DCT components rather than individual components.

SoftCast exploits these two properties to efficiently compress the data by transmitting only the non-zero DCT components. This compression is very efficient and has no impact on the energy in a frame. However, it requires the encoder to send a large amount of metadata to the decoder to inform it of the locations of the discarded DCT components.

To reduce the metadata, SoftCast groups nearby spatial DCT components into *chunks*, as shown in Fig. 3-2c. The default chunk in our implementation is 44x30x1 pixels, (where 44×30 is chosen based on the SIF video format where each frame is 352×240 pixels). Note that SoftCast does not group temporal DCT components because typically only a few structures in a frame move with time, and hence most temporal components are zero, as in Fig. 3-2c. SoftCast then makes one decision for all DCT components in a chunk, either retaining or discarding them. The clustering property of DCT components allows SoftCast to make one decision per chunk without compromising the compression it can achieve. As before, the SoftCast encoder still needs to inform the decoder of the locations of the non-zero chunks, but this overhead is significantly smaller since each chunk represents many DCT components (the default is 1320 components/chunk). SoftCast sends this location information as a bitmap. Again, due to clustering, the bitmap has long runs of consecutive retained chunks, and hence can be efficiently compressed using run-length encoding.

The previous discussion assumed that the source has enough bandwidth to transmit all the non-zero chunks over the wireless medium. What if the source is bandwidth constrained? It will then have to judiciously select non-zero chunks so that the transmitted stream can fit in the available bandwidth, and still be reconstructed with the highest quality. SoftCast selects the transmitted chunks so as to minimize the

reconstruction error at the decoder:

$$err = \sum_i (\sum_j (x_i[j] - \hat{x}_i[j])^2), \tag{3.1}$$

where $x_i[j]$ is the original value for the j^{th} DCT component in the i^{th} chunk, and $\hat{x}_i[j]$ is the corresponding estimate at the decoder. When a chunk is discarded, the decoder estimates all DCT components in that chunk as zero. Hence, the error from discarding a chunk is merely the sum of the squares of the DCT components of that chunk. Thus, to minimize the error, SoftCast sorts the chunks in decreasing order of their energy (the sum of the squares of the DCT components), and picks as many chunks as possible to fill the bandwidth.

Note that bandwidth is a property of the source, (e.g., a 802.11 channel has a bandwidth of 20 MHz) independent of receiver, whereas SNR is a property of the receiver and its channel. As a result, discarding non-zero chunks to fit the source bandwidth does not prevent each receiver from getting a video quality commensurate with its SNR.

Two points are worth noting about the used compression.

- SoftCast can capture correlations across frames while avoiding motion compensation and differential encoding. It does this because it performs a 3D DCT, as compared to the 2-D DCT performed by MPEG. The ability of the 3D DCT to compact energy across time is apparent from Fig. 3-2b where the values of the temporal DCT components die quickly (i.e., in Figs. 3-2b, the planes in the back are mostly black).

- The main computation performed by SoftCast's compression is the 3D DCT, which is $O(K \log(K))$, where K is the number of pixels in a GoP. A variety of efficient DCT implementations exist both in hardware and software [25,70].

- Finally, it is possible to replace 3D DCT with other 3D decorrelation transforms, such as 3D Wavelets [106]. I have experimented with both 3D DCT and 3D Wavelets and found them to be comparable, with 3D DCT showing better

clustering of non-zero components.

3.3.2 Error Protection

Traditional error protection codes transform the real-valued video data to bit sequences. This process destroys the numerical properties of the original video data and prevents us from achieving our design goal of having the transmitted digital samples scale linearly with the the pixel values. Thus, SoftCast develops a novel approach to error protection that is aligned with its design goal. SoftCast's approach is based on scaling the magnitude of the DCT components in a frame. Scaling the magnitude of a transmitted signal provides resilience to channel noise. To see how, consider a channel that introduces an additive noise in the range ± 0.1. If a value of 2.5 is transmitted directly over this channel, (e.g., as the I or Q of a digital sample), it results in a received value in the range $[2.4 - 2.6]$. However, if the transmitter scales the value by $10x$, the received signal varies between 24.9 and 25.1, and hence when scaled down to the original range, the received value is in the range $[2.51 - 2.49]$, and its best approximation given one decimal point is 2.5, which is the correct value. However, since the hardware has a fixed power budget, scaling up and therefore expending more power on some signal samples translates to expending less power on other samples. SoftCast's optimization finds the optimal scaling factors that balance this tension.

Again, we operate over chunks, i.e., instead of finding a different scaling factor for each DCT component, we find a single optimal scaling factor for all the DCT components in each chunk. To do so, we model the values $x_i[j]$ within each chunk i as random variables from some distribution $\mathcal{D}_i$. We remove the mean from each chunk to get zero-mean distributions and send the means as metadata. Given the mean, the amount of information in each chunk is captured by its variance. We compute the variance of each chunk, λ_i, and define an optimization problem that finds the per-chunk scaling factors such that GoP reconstruction error is minimized. In the appendix, we show:

Lemma 3.3.1. *Let $x_i[j], j = 1 \ldots N$, be random variables drawn from a distribution*

$\mathcal{D}_i$ with zero mean, and variance λ_i. Given a number of such distributions, $i = 1 \ldots C$, a total transmission power P, and an additive white Gaussian noise channel, the linear encoder that minimizes the mean square reconstruction error is:

$$\begin{aligned} u_i[j] &= g_i x_i[j], \quad \textit{where} \\ g_i &= \lambda_i^{-1/4} \left(\sqrt{\frac{P}{\sum_i \sqrt{\lambda_i}}} \right). \end{aligned}$$

Note that there is only one scaling factor g_i for every distribution $\mathcal{D}_i$, i.e., one scaling factor per chunk. The output of the encoder is a series of coded values, $u_i[j]$, as defined above. Further, the encoder is linear since DCT is linear and our error protection code performs linear scaling.

3.3.3 Resilience to Packet Loss

Next, we assign the coded DCT values to packets. However, as we do so, we want to maximize SoftCast's resilience to packet loss. Current video design is fragile to packet loss because it employs differential encoding and motion compensation. These schemes create dependence between packets, and hence the loss of one packet can cause subsequent correctly received packets to become undecodable. In contrast, SoftCast's approach ensures that all packets equally important. Hence, there are no special packets whose loss causes disproportionate video distortion.

A naive approach to packetization would assign chunks to packets. The problem, however, is that chunks are not equal. Chunks differ widely in their energy (which is the sum of the squares of the DCT components in the chunk). Chunks with higher energy are more important for video reconstruction, as evident from equation 3.1. Hence, assigning chunks directly to packets causes some packets to be more important than others.

SoftCast addresses this issue by transforming the chunks into equal-energy *slices*. Each SoftCast slice is a linear combination of all chunks. SoftCast produces these

slices by multiplying the chunks with the Hadamard matrix, which is typically used in communication systems to redistribute energy [6,84]. The Hadamard matrix is an orthogonal transform composed entirely of +1s and -1s. Multiplying by this matrix creates a new representation where the energy of each chunk is smeared across all slices.[1]

We can now assign slices to packets. Note that, a slice has the same size as a chunk, and depending on the chosen chunk size, a slice might fit within a packet, or require multiple packets. Regardless, the resulting packets will have equal energy, and hence offer better packet loss protection.

The packets are delivered directly to the PHY (via a raw socket), which interprets their data directly as the digital signal samples to be sent on the medium, as described in §3.5.

3.3.4 Metadata

In addition to the video data above, the encoder sends a small amount of metadata to assist the decoder in inverting the received signal. Specifically, the encoder sends the mean and the variance of each chunk, and a bitmap that indicates the discarded chunks. The decoder can compute the scaling factors, i.e., g_i's, from this information. As for the Hadamard and DCT matrices, they are well known and do not need to be transmitted. The bitmap of chunks is compressed using run length encoding as described in §3.3.1, and all metadata is further compressed using Huffman coding. The total metadata in our implementation after adding a Reed-Solomon code is 0.014 bits/pixel, i.e., its overhead is insignificant.

The metadata has to be delivered correctly to all receivers. To protect the metadata from channel errors, we send it using BPSK modulation and half rate convolutional code, which are the modulation and FEC code corresponding to the lowest 802.11 bit rate. To ensure that the probability of losing metadata because of packet loss is very low, we spread the metadata across all packets in a GoP. Thus, each of

[1]Hadamard multiplication has an additional benefit which is to whiten the signal reducing the peak to average power ratio (PAPR).

SoftCast's packets starts with a standard 802.11 header followed by the metadata then the coded video data. (Note that different OFDM symbols in a packet can use different modulation and FEC code. Hence, we can send the metadata and the SoftCast video data in the same packet.) To further protect the metadata we encode it with a Reed-Solomon code that can tolerate a loss rate up to 50%. The code uses a symbol size of one byte, a block size of 1024, and a redundancy factor of 50%. Thus, even with 50% packet erasure, we can still recover the metadata fully correctly. This is a high redundancy code but since the metadata is very small, we can afford a code that doubles its size.

3.3.5 The Encoder: A Matrix View

We can compactly represent the encoding process of a GoP as matrix operations. Specifically, we represent the DCT components in a GoP as a matrix X where each row is a chunk. We can also represent the final output of the encoder as a matrix Y where each row is a slice. The encoding process can then be represented as

$$Y = HGX \tag{3.2}$$

$$= CX \tag{3.3}$$

where G is a diagonal matrix with the scaling factors, g_i, as the entries along the diagonal, H is the Hadamard matrix, and $C = HG$ is simply the encoding matrix.

3.4 Decoder

At the receiver, and as will be described in §3.5, for each received packet, the PHY returns the list of coded DCT values in that packet (and the metadata). The end result is that for each value $y_i[j]$ that we sent, we receive a value $\hat{y}_i[j] = y_i[j] + n_i[j]$, where $n_i[j]$ is random noise from the channel. It is common to assume the noise is additive, white and Gaussian. While this is not exact, it works reasonably well in practice.

The goal of the SoftCast receiver is to decode the received GoP in a manner that minimizes the reconstruction errors. We can write the received GoP values as

$$\hat{Y} = CX + N,$$

where $\hat{Y}$ is the matrix of received values, C is the encoding matrix from Eq. 3.2, X is the matrix of DCT components, and N is a matrix where each entry is white Gaussian channel noise.

Without loss of generality, we can assume that the slice size is small enough that a slice fits within a packet, and hence each row in $\hat{Y}$ is contained in a single packet. If the slice size is larger than the packet size, then each slice consists of more than one packet, say, K packets. The decoder simply needs to repeat its algorithm K times. In the i^{th} iteration ($i = 1 \ldots K$), the decoder constructs a new $\hat{Y}$ where the rows consist of the i^{th} packet from each slice.[2] For the rest of our exposition, therefore, we will assume that each packet contains a full slice.

The receiver knows the received values, $\hat{Y}$, and can construct the encoding matrix C from the metadata. It then needs to compute its best estimate of the original DCT components, X. The linear solution to this problem is widely known as the Linear Least Square Estimator (LLSE) [57]. The LLSE provides a high-quality estimate of the DCT components by leveraging knowledge of the statistics of the DCT components, as well as the statistics of the channel noise as follows:

$$X_{LLSE} = \Lambda_x C^T (C\Lambda_x C^T + \Sigma)^{-1} \hat{Y}, \tag{3.4}$$

where:

- X_{LLSE} refers to the LLSE estimate of the DCT components.
- C^T is the transpose of the encoder matrix C.
- Σ is a diagonal matrix where the i^{th} diagonal element is set to the channel noise

[2]Since matrix multiplication occurs column by column, we can decompose our matrix $\hat{Y}$ into strips which we operate on independently.

power experienced by the packet carrying the i^{th} row of $\hat{Y}$. The PHY has an estimate of the noise power in each packet, and can expose it to the higher layer.

- Λ_x is a diagonal matrix whose diagonal elements are the variances, λ_i, of the individual chunks. Note that the λ_i's are transmitted as metadata by the encoder.

Consider how the LLSE estimator changes with SNR. At high SNR (i.e., small noise, the entries in Σ approach 0), Eq. 3.4 becomes:

$$X_{LLSE} \approx C^{-1}Y \tag{3.5}$$

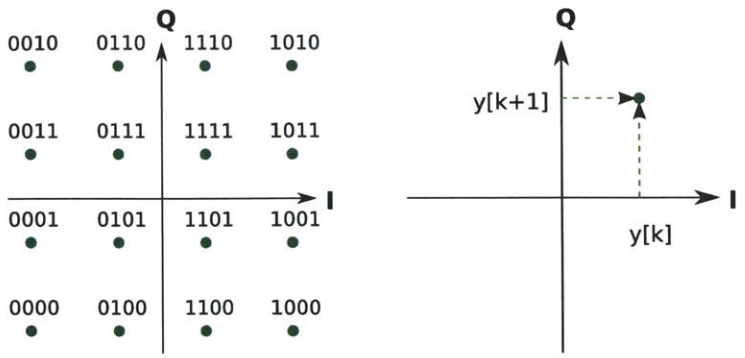
Thus, at high SNR, the LLSE estimator simply inverts the encoder computation. This is because at high SNR we can trust the measurements and do not need to leverage the statistics, Λ, of the DCT components. In contrast, at low SNR, when the noise power is high, one cannot fully trust the measurements and hence it is better to re-adjust the estimate according to the statistics of the DCT components in a chunk.

Once the decoder has obtained the DCT components in a GoP, it can reconstruct the original frames by taking the inverse of the 3D DCT.

3.4.1 Decoding in the Presence of Packet Loss

We note that, in contrast to conventional 802.11, where a packet is lost if it has any bit errors, SoftCast accepts all packets. Thus, packet loss occurs only when the hardware fails to detect the presence of a packet, e.g., in a hidden terminal scenario.

Still, what if a receiver experiences packet loss? When a packet is lost, SoftCast can match it to a slice using the sequence numbers of received packets. Hence the loss of a packet corresponds to the absence of a row in Y. Define Y_{*i} as Y after removing the i^{th} row, and similarly C_{*i} and N_{*i} as the encoder matrix and the noise vector after



(a) 16-QAM (b) SoftCast

Figure 3-3: **Mapping coded video to I/Q components of transmitted signal.** For example, to transmit the bit sequence 1010, the traditional PHY maps it to the complex number corresponding to the point labeled 1010. In contrast, SoftCast's PHY treats pairs of coded values as the real and imaginary parts of a complex number.

removing the i^{th} row. Effectively:

$$\hat{Y}_{*i} = C_{*i}X + N_{*i}. \tag{3.6}$$

The LLSE decoder becomes:

$$X_{LLSE} = \Lambda_x C_{*i}^T (C_{*i}\Lambda_x C_{*i}^T + \Sigma_{(*i,*i)})^{-1}\hat{Y}_{*i}. \tag{3.7}$$

Note that we remove a row and a column from Σ. Eq. 3.7 gives the best approximation of Y when a single packet is lost. The same approach extends to any number of lost packets. Thus, SoftCast's approximation degrades gradually as receivers lose more packets, and, unlike MPEG, there are no special packets whose loss prevents decoding.

3.5 Interfacing with the PHY Layer

Traditionally, the PHY layer takes a stream of bits and codes them for error protection. It then modulates the bits to produce real-valued digital samples that are transmitted on the channel. For example, 16-QAM modulation takes sequences of 4 bits and maps each such sequence to a complex number as shown in Fig. 3-3a. The real and imaginary parts of these complex numbers produce the real-valued I and Q

components of the transmitted signal.

In contrast to existing wireless design, SoftCast's codec outputs real values that are already coded for error protection. Although SoftCast's codewords are protected from additive noise, a wireless channel introduces a plethora of non-additive effects such as inter-symbol interference, frequency-selective attenuation, timing, sampling, phase, and frequency offsets. In today's systems, the PHY layer deals with those phenomena by estimating and cancelling their effect on the received signal. Rather than reimplementing this functionality in SoftCast, the current PHY can be reorganized to create RawPHY which accepts real-valued inputs while protecting from the non-additive effects. In a simplified view, RawPHY can be created by bypassing the components of PHY that operate on bits: forward error correction and digital modulation (mapping). Thus, we can directly map pairs of SoftCast coded values to the I and Q digital signal components, as shown in Fig. 3-3b.[3] The next chapter discusses the design and implementation of RawPHY in depth.

[3]An alternative way to think about SoftCast is that it is fairly similar to the modulation in 802.11 which uses 4QAM, 16QAM, or 64QAM, except that SoftCast uses a very dense 64K QAM.

Chapter 4

RawPHY: Interface to a Practical Wireless Channel

Although SoftCast takes care of mapping of the pixels to real-valued channel samples, those samples are not suitable for direct transmission over the wireless channel. In this chapter, I discuss the design of RawPHY which provides SoftCast with a suitable *waveform* interface to the wireless channel while protecting its codewords from non-additive effects of the channel. RawPHY also allows integration of SoftCast alongside traditional digital data transmission.

4.1 Channel Coding in Practice

Shannon's source-channel separation principle promotes a layered system design. The *physical* layer, or PHY, deals with the specifics of the channel to the extent that the higher layers do not need to handle bit errors. Conversely, the *application* layer deals with the domain-specific issues, such as video compression and streaming formats keeping the lower layers oblivious to the transferred payload. Thus, the communication system design today shields the designers of the application protocols from issues specific to the physical medium such as how the bits are represented on the channel, how the packets are detected at the receiver, or how the channel noise affects the signal. Such separation of concerns greatly reduces the complexity of the overall

system design.

In reality, the setting of Shannon's result is idealized: it assumes that the channel conditions and source distribution does not change and permits the source and channel coders to operate on unbounded spans of source and channel signals. In practice, the PHY can only deal with channel signals of limited length. Hence it does not guarantee error-free stream but instead delivers packets free from bit errors, although some of the packets can be lost. The link and medium access control (MAC) layers above it typically mask some of the packet erasures, although ultimately the typical *best effort* system design does not guarantee packet delivery even at the network layer, leaving the issue of reliability up to the communication end-points. Such design is embodied in the Internet and local-area computer networks (LAN).

There are several reasons for leaving end-to-end reliability to the end points. The *end-to-end argument* [80] dictates that the network could only resort to improve reliability for the sake of performance but cannot effectively hide all failures from the application. In the context of streaming, reliability becomes secondary to the primary goal of on-time delivery of content. For example, when streaming a video, it is better to skip a frame than stall until the data reliably arrives. To support streaming applications, the current architecture allows the application to express a "wish" that its data should be delivered with minimum delay but higher chance of loss. For example, a 802.11 PHY supporting Wireless Media Extensions (WME) would use smaller retransmission count for data tagged as video.

Different applications have distinct preferences in the reliability vs. latency trade-off, so this contract is widely supported in the modern wireless architecture. However, the current abstraction still imposes an all-or-nothing contract on application data packets. Even if this contract is relaxed to that of a noisy bit-pipe (e.g., by ignoring the CRC check on the received packets), such design is merely a workaround and requires the application end-points to apply their own FEC. Although the application can create layers of bit-pipes of varying importance with an extended contract – "pass as many correct bits as possible, according to priority" – this not only no longer isolates the application and PHY design, but as I show in the next chapter, can also

be outperformed by a simpler design, as in SoftCast, in which the application has control over channel representation of its data.

The new PHY must grant the higher layers (i.e., the application) full control not only on the rate of the channel code, but also the specific allocation of codewords within the signal space. Thus let us replace the bit-pipe abstraction with a *waveform interface*. However, removing the PHY layer completely would unnecessarily expose the designers of the application to the heterogeneity of the transceiver hardware and inevitably increase complexity. Instead I pose the question: Once binary coding and digital modulation are removed from the PHY, what core functionality should the remaining RawPHY have? My proposal is that RawPHY should deal with as many effects of the channel as possible without sacrificing the potential gains of the redefined contract. For this purpose I introduce the notion of *universality* of particular channel coding functionality with respect to any channel coding applied in the application. In many cases, such universality comes at the price of efficiency. I discuss this trade-off for three such aspects of the PHY: masking frequency-selective fading, reduction of peak-to-average-power ratio (PAPR), and space-time block coding.

In the second part of this chapter, I describe a prototype design and implementation of RawOFDM, an OFDM-based RawPHY layer, within the GNURadio software radio framework.

4.2 Design

In this section, I consider the requirements for a practical design of a PHY layer that would provide a waveform interface. In attempt to specify the minimum and maximum functionality of a RawPHY layer, I consider the functional specification of a PHY layer in the current organization of communication systems and discuss which aspects of this specification can still be provided by a RawPHY layer. This discussion is structured around the concept of layering of channel coding techniques within the PHY layer. In a trivial design, one could simply "peel off" some of those layers until the effective channel has waveform behavior. Later, I show that some core

PHY functionality can be preserved within RawPHY thus reducing the complexity burden of the applications.

4.2.1 A Stack of Layered Channels

The key observation leading to our design of RawPHY is that the modern PHY is internally layered [9]. That is, we can organize the pipeline of processing blocks of the transmitter and receiver chains into layers by matching the complementary transmitter and receiver blocks. Each layer defines an end-to-end channel which takes the input of the transmitter block, performs some transformation, and delivers output at the corresponding receiver block. The layer is in essence a channel code applied to the channel provided by the lower layer. In principle, the channel code masks the effects of the lower channel but possibly introduces new effects to the upper channel. For instance, digital modulation wraps a waveform channel with additive noise into a new binary channel. This new layer has its own characteristics, such as correlation in the bit error distribution. This, in turn, can be masked via interleaving, yielding a channel which can be modeled as memoryless BSC, and so on. Thus, in a typical PHY layer, the effective channel code is composed of several concatenated codes. This approach promotes component-based development which benefits both the theoretical analysis and hardware implementation.

Consider the conceptual organization of an idealized wireless PHY illustrated in Fig. 4-1. The processing blocks on the right convert the lower channel type to the upper channel type. We focus on the characterization of the channels in between the layers.

Linear Time Invariant (LTI): The baseband signal is an unstructured stream of digital samples. It has an unknown impulse response and there is no synchronization between the transmitter and receiver. However, with high accuracy, if we ignore frequency offset in carrier or sampling, the channel can be modelled as linear-time invariant.

LTI Symbols: The transmitter adds preambles which are known symbols and allow

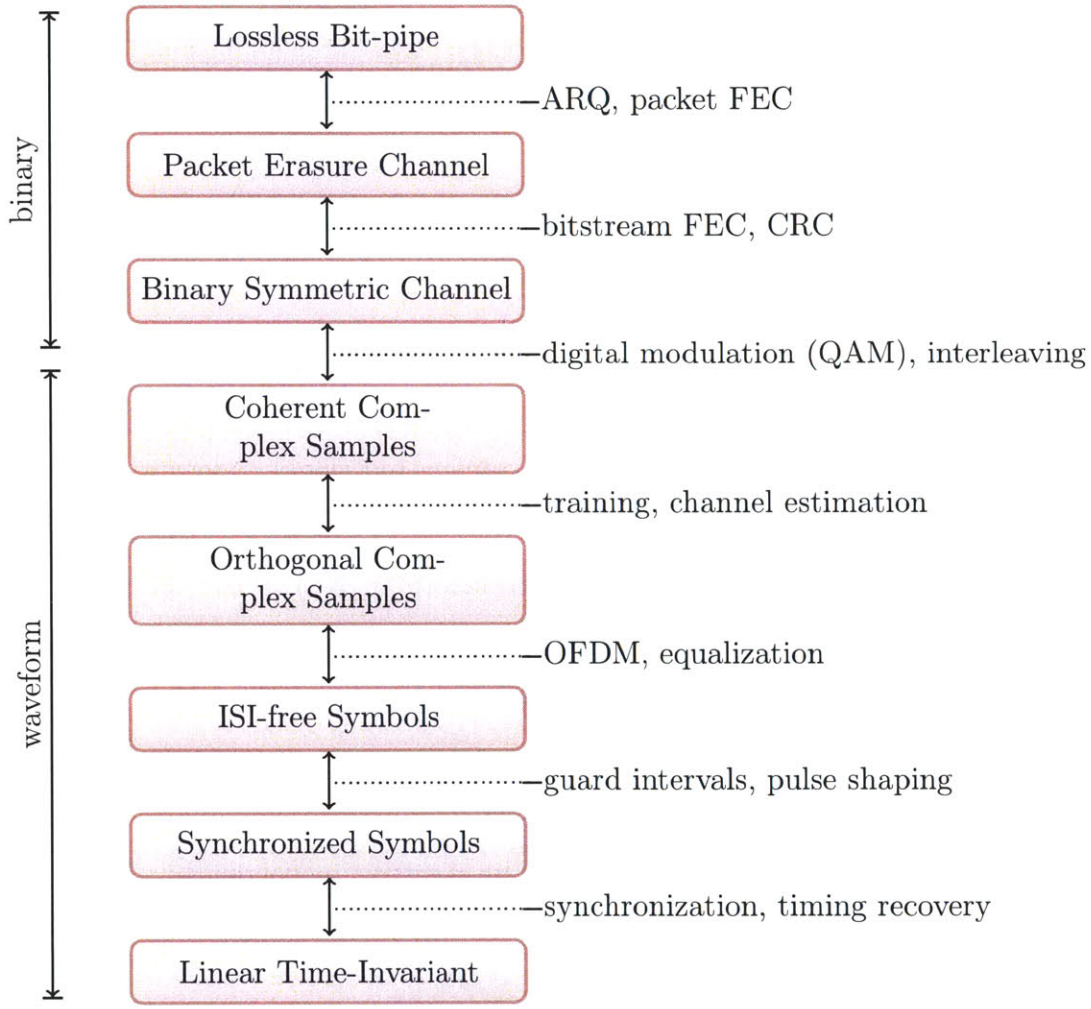


Figure 4-1: **Layered PHY.** Conceptual organization of a typical coherent wireless PHY layer into sub-layers and the corresponding effective channels. The processing blocks on the right convert the lower channel type to the upper channel type. This separation of processing into layers is idealized and in most practical realizations there are loopy interactions between processing blocks.

the receiver to identify the beginning of the frame. Moreover, the receiver can also establish timing, i.e., symbol boundaries. However, the symbols are not quite independent. This is because the channel is band-limited and so an impulse is spread over many samples possibly leaking into the adjacent symbols.

Orthogonal Symbols: Inter-symbol interference (ISI) can be addressed at the transmitter or the receiver. The transmitter can apply pulse shaping or insert guard intervals (with cyclic prefix) between symbols to ensure that the channel response of one symbol does not spread into the next. The receiver can also neutralize ISI to some extent by canceling (subtracting) the estimated channel response.

Orthogonal Samples: Since the samples within a symbol are still spread by the channel throughout the symbol, the transmitted signal codewords need to be discernible. Orthogonal frequency division multiplexing (OFDM) simplifies this task by applying the digital Fourier transform (DFT) to the channel. Specifically, the transmitter treats the samples in the input symbol as the frequency domain and applies IDFT to determine the time-domain signal to be transmitted on the channel. The receiver applies DFT to retrieve the original samples in frequency domain. The end-to-end channel above this layer guarantees that the complex samples in each symbol can be treated independently. To guarantee this property in practical OFDM systems, the receiver must detect and correct any offset in carrier frequency.

Coherent Complex Samples: Although orthogonal, the stream of complex numbers is subject to, possibly non-flat, frequency response of the underlying LTI channel. In the frequency domain, this manifests as a multiplicative complex factor applied to each frequency bin. If the channel is truly time invariant, this factor is the same across the same bin in different symbols. The decoder must unambiguously estimate and invert this factor to allow coherent reception.[1]

[1]Notably, a number of communication systems employ differential encoding and do not depend on coherent reception, since an LTI channel preserves relative phase [91].

This is again accomplished by training, i.e., the known preamble inserted by the transmitter is used at the receiver to estimate the factor for each frequency bin. In practice, the channel could be slowly fading and so the channel response could change from symbol to symbol. In particular, residual frequency offset would result in the phase of all factor changing from one symbol to the next. This phase can be tracked by fine-grained training using injected pilot tones.

Additive Noise Channel: After proper equalization, the waveform channel above this layer resembles AWGN in that the noise is additive.[2] However, the noise is not white as the frequency bins experiencing stronger attenuation will show higher noise variance after equalization. If fading is slow enough, this channel state information (CSI) can be communicated to the upper layer which can then adjust the code rate to the SNR of individual bins. If such CSI is not available on time, or if simplicity is preferred, this effect can be masked by interleaving, i.e., decorrelation.

(Quasi) Binary Symmetric Channel: Digital modulation (e.g., QAM) wraps a waveform channel with additive noise into a new, binary channel by reducing the number of codewords in the codebook to a finite constellation. Closer scrutiny reveals that the bit errors on the resulting channel are not independent nor identically distributed. As each constellation point encodes multiple bits, one symbol error (when noise vector is large enough to confuse one point with another) can result in error in more than one of those bits. Thus errors in the bits within one block are correlated. Furthermore, depending on the mapping of specific bit sequences to the constellation points, some bits in a block can be protected more than others. As before, interleaving and scrambling decorrelates those errors and the result can be modeled a a binary symmetric channel (BSC).

Packet Erasure Channel: Forward error correcting (FEC) coding reduces the probability of bit error at the expense of bit rate. When the probability of an error in a frame is very low, the PHY can mask bit errors completely by adding an

[2]Although if the equalization is inaccurate, the channel distortion is multiplicative.

error detection code such as CRC-32. If the receiver detects a bit error, it drops the frame. Thus, the channel above this layer is a packet erasure channel.

Lossless Bit-pipe: Although packet erasure is the standard contract on the network layer, the end-to-end performance can be improved by masking some of the erasures from the application. Modern WLAN employs link-layer retransmission (i.e., automatic repeat request, ARQ) to greatly reduce the number of erasures observed by the applications. The rationale behind this design is that if the packet loss is due to channel fading, as opposed to congestion, then the erasure should be masked from the transport layer to avoid back-off. Instead, the link layer can adjust the code rate and exploit temporal diversity to improve packet delivery. In an idealized model, the channel above this layer is a lossless bit-pipe.

4.2.2 Interactions and Universality of Channel Codes

Notably, this separation of processing into layers is idealized and in most practical realizations there are loopy interactions between processing blocks. Generally, a maximum likelihood (ML) decoder of a channel code aims to identify the most likely transmitted codeword given the received signal. Thus, if the decoder knows the probability distribution of all the codewords in the codebook p_X, and it knows the distribution of the channel $p_{Y|X}$, then given channel output y the ML-decoded output $\hat{x}$ is computed as:

$$\hat{x} = \arg\max_x Pr(x|y) = \arg\max_x p_{Y|X}(y|x)p_X(x)$$

However, the decoders knowledge of p_X is limited by isolation from any outer channel code. Specifically, X is the output of some functional transformation of the encoder's input M, i.e. $f : M \to X$. If the distribution p_M is known to the decoder, then it can also determine $p_X(x) = p_M(f^{-1}(x))$. However, an error-correcting code must necessarily make the blocks or sequences of M that correspond to valid code-words more likely than non-codeword blocks in order to introduce some distance between the valid codewords. For practical simplicity, the decoder of X is often designed under

the assumption that p_M is uniform (or otherwise simply modeled without complete knowledge of the codebook of M).

A decoder that makes a final decision regarding X (and thus $M = f^{-1}(X)$) without knowledge of p_M, is called *hard-decision* and is generally characterized by a significant performance loss when compared to the channel capacity. *Soft-decision* decoding operates without knowledge of p_M but defers the final decision regarding X. The output of a soft-output decoder is a measure of confidence in the proposed ("most likely") X. The outer code can then essentially correct codewords of low confidence. Hence soft-decision decoding alleviates this performance loss while preserving most of the isolation and thus keeping the complexity of the system low. In result, although the current boundary between binary and waveform interfaces is nominally defined by the digital modulation as shown in Fig. 4-1, soft-decision decoding requires soft-value output from the digital modulation decoder, and hence the BSC is effectively replaced with a soft-value channel. Both the Viterbi and BCJR algorithms for trellis codes have soft-input soft-output variants.

However, in many cases such decomposition proves difficult. For instance, reducing inter-symbol interference (ISI) via a decision-feedback equalizer (DFE) (which is essentially interference cancellation) requires the involvement of the higher layers (i.e., outer code such as digital modulation) to re-encode the samples in the previous symbol so that they can be subtracted from the received samples in the current symbol. Similarly, frequency equalization for orthogonal samples (after OFDM) could take into account the output from the constellation demapper which operates in a higher layer. If the phase noise is low, the constellation demapper can yield highly accurate estimate of current phase offset. As the last example, decoding of the digital superposition code described in Section 5.3.1 requires successive layer cancellation and thus for optimal performance must involve *all* of outer channel coding.

The layered PHY is essentially a series of concatenated channel codes, and for the purpose of this discussion, an inner channel code is considered *universal* with respect to the outer code if both its encoder and decoder can operate in a manner oblivious to any outer code. In contrast, if a channel code is not universal, then joint decoding

or encoding is necessary for optimal performance. As illustrated above, such universality is generally unattainable for error-correcting codes. However, there is a number of components that are universal in practice. Examples of universal blocks include: preamble detection, symbol and sample orthogonalization at the transmitter, scrambling and interleaving, preamble- and pilot-based channel estimation, and orthogonal space-time codes (which are typically applied above orthogonal samples).

4.2.3 A Waveform PHY

Although prior work proposed to expose non-binary confidence information at the receiver [48,52,98,101], to reap full benefits of joint source-channel coding, RawPHY must present a waveform interface at the transmitter. To expose a waveform interface to the channel, the processing blocks which operate on bits need to be removed from the PHY layer. In doing so, the remaining blocks must remain oblivious to any channel codes that could be applied above the newly exposed waveform interface. Naturally, by above definition, any universal blocks are suited to remain within the PHY thus reducing the complexity burden of the higher layers. However, to serve its purpose, RawPHY must exclude the channel coding blocks that are subject to adaptation to channel conditions, i.e., digital modulation, error-correcting codes, etc. Notably, other aspects of the PHY, such as baseband channel width (i.e., sampling frequency), symbol length, anti-ISI guard interval, number of preambles or pilots, can also be adjusted depending on channel conditions, such as multi-path delay spread. However, exploiting this adaptivity in a broadcast setting, where receivers observe potentially diverse channel conditions, is rather difficult and beyond the scope of this thesis.

The proposed waveform PHY is to provide a channel of *coherent orthogonal real samples*. Consider that the binary interface is simple: the conventional PHY accepts a packet of bits. In contrast, RawPHY accepts a packet of real (fix-point or floating-point) numbers. The typical complex samples are treated as pairs of reals for the sake of design simplicity of the channel codes employed by the higher layers. More precisely, non-orthogonal effects such as phase noise should be masked from the higher

layers.

Thus RawPHY is comprised of the channel coding blocks necessary to create such channel: detection, synchronization, orthogonalization, equalization, etc. Given the universality requirement, those blocks must not depend on any channel coding applied on top of the waveform interface. In particular, RawPHY cannot assume any structure in the signal generated by the higher layer. However, some additional specification is needed.

Signal power: Since the real field is not finite, the distribution over the field becomes of importance. Specifically, if the real samples were directly converted to baseband, the power of the distribution, $E[X^2]$, will determine the power output of the transmitter. On one hand, the EM power is subject to government regulation. On the other hand, the hardware cannot support unbounded signal and will introduce non-linearities in the form of clipping. RawPHY could enforce the power limit internally, along the lines of bit-stuffing or character escaping in conventional link layers. Alternatively, RawPHY would simply assume that the input power is normalized since the negative side-effects of non-conformance provide an incentive to the higher layer to respect this requirement. Similar reasoning suggests that the higher layer would ensure that the input signal has zero-mean (since the mean does not carry much information but substantially increases total power).

Signal distribution: Overall signal power is not the only relevant statistic. In particular, the PHY might have to clip values of high magnitudes due to fixed precision of baseband digital processing, or limited range of the D/A and A/D converters. However, requiring a strict conformance with a particular distribution is excessive. Instead, RawPHY performs internal *whitening* of the signal in a manner similar to the conventional pseudo-random scrambling of the bit-stream before the digital FEC. In the next section, I discuss the details of this procedure and its effect on such properties of the signal as peak-to-average power ratio (PAPR) and channel noise profile.

4.2.4 Additional Facilities for the Application

In addition to the bare waveform interface, it is worthwhile to consider other information that the PHY could provide to the application.

Channel state information: Some higher layer components can leverage the current measured statistics of the channel such as the signal-to-noise ratio (SNR). This is usually done by using known symbols such as preambles or pilot tones but can also be done by using known structure in the signal, such as the QAM constellation. Although the application can implement this, if the PHY is already performing channel estimation for equalization, it can determine the resulting statistic of interest and make it available to application requests.

Signal identification: To determine the adaptive parameters of the channel coding (i.e., constellation, FEC code rate, etc.), the conventional PHY internally encodes a SIGNAL field using a fixed channel code. This field instructs the receiver how to decode the payload of the packet. In RawPHY, there is no essential need for such information as the application can take care of this issue. However, since there could be multiple applications sharing the wireless device, providing some signal identification facilities in RawPHY could allow demultiplexing the different applications. Such signal identifier would correspond to the TYPE field in the Ethernet protocol. Another way to accomplish this demultiplexing without a reliably encoded SIGNAL field is to allow the applications (recall, this term includes any higher layer users of the PHY, e.g., the MAC layer) to register a signature with RawPHY. This signature, if sufficiently uncorrelated from signatures of other applications could allow quick rejection of signals that are not relevant to the application. In fact, a known pseudo-random preamble is commonly employed in conventional PHY to detect frames of interest from other users of the shared RF spectrum. Consider for example, 802.11, Bluetooth, Zigbee, and other users of the ISM 2.4GHz band.

Protocol Encapsulation: Finally, although beyond the scope of this thesis, consider the case when the waveform payload is decoded at a destination beyond the wireless receiver. A hybrid packet could be encoded partially using the conventional digital channel codes and partially determined by the end-application. This corresponds to protocol encapsulation in the common network protocol stack with the difference that the digital channel codes are only applied to a part of the payload. Alternatively, digital coding can be provided in suite with RawPHY The application (i.e., using a socket abstraction) can then decide whether to use a standard digital channel, or to have its data transmitted without FEC and digital modulation.

4.3 RawOFDM

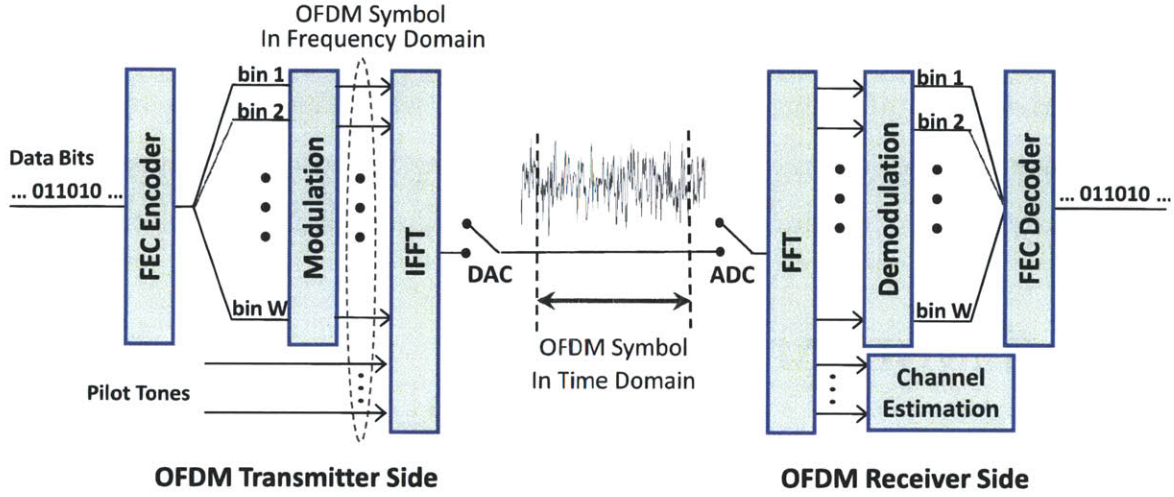
In this section, I describe a practical implementation of RawPHY which uses OFDM to create an effective channel of coherent real signal samples which forms the basis of the waveform interface. This approach leverages the sub-layering in the modern PHY discussed in the previous section which makes RawOFDM, in essence, a small modification of the current OFDM-based design. In fact, when combined with conventional digital channel codes, RawOFDM is equivalent (modulo specific parameters) to the modern PHY as embodied in 802.11a/g/n, 802.16 or DVB-T/T2/H.

4.3.1 Overview

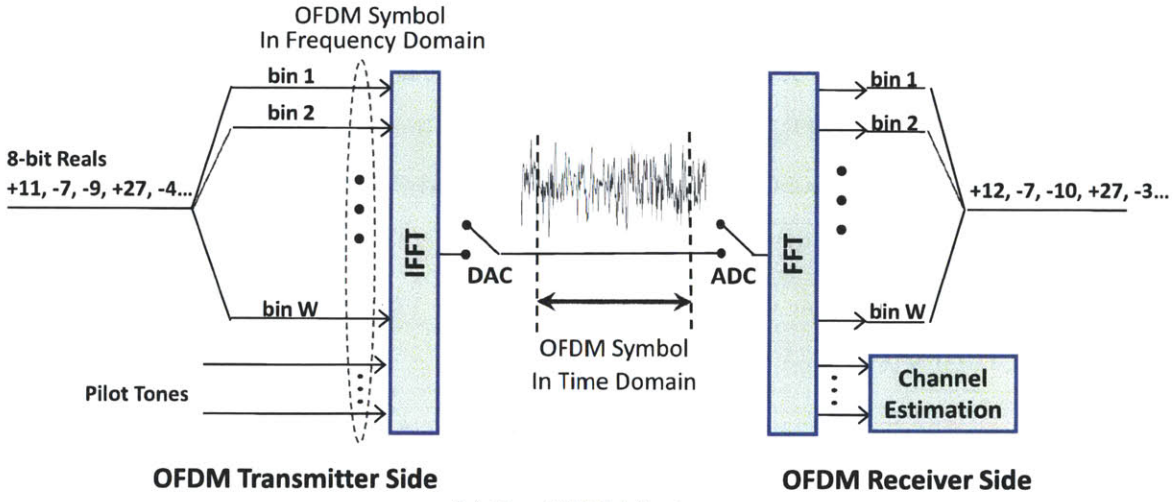
In principle, RawOFDM starts with the conventional OFDM system as illustrated in the simplified schematic in Fig. 4-2a. Each OFDM symbol is converted into frequency domain via IDFT and divided into a number of frequency bins. Each bin in each symbol carries one complex signal sample, i.e., a pair of in-phase and quadrature components (I/Q).

The typical OFDM system uses QAM to map chunks of bits to complex constellation symbols. Before that, the bits are interleaved and encoded using FEC.[3] In comparison, RawOFDM bypasses FEC and modulation, presenting a waveform

[3]For more details about those concepts see Sections 2.2.2 and 2.2.4.

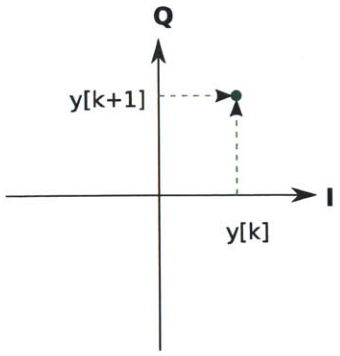


(a) Typical OFDM System



(b) RawOFDM System

Figure 4-2: **RawOFDM in essence.** Adapting typical OFDM-based PHY to RawPHY can be accomplished by moving modulation and FEC blocks to the higher layer.



(a) raw

Figure 4-3: **Mapping raw input to the I/Q components of transmitted signal.** RawOFDM treats pairs of coded values as the real and imaginary parts of a complex number.

channel interface to higher layers, as shown in the schematic in Fig. 4-2b. The significant overlap between RawOFDM and the standard OFDM makes it feasible to support a flexible interface offering both the bit-pipe and raw options to the application. Streaming media applications can choose the raw option if they want to prevent the cliff effect, while traditional file transfer applications can rely on the standard PHY and link layers to provide core reliability.

In the raw mode, the PHY does not know whether a packet contains video, audio, or other source signals. From the perspective of the PHY, it receives a stream of values that it needs to put on the raw OFDM channel. It assigns a pair of values to each data bin, mapping one to the in-phase, and the other to the quadrature component, of the signal. This is illustrated in Fig. 4-3. As before, these signals are then input to an IFFT module to generate a time-domain signal which is transmitted across the channel. At the receiver, the time signal is sent to an FFT to retrieve the values in each of the OFDM bins. These values are equalized using standard OFDM processing. At this point, the traditional OFDM pipeline of demodulation and FEC is bypassed, and the raw values are transmitted to the receive socket. Note that these values are not subjected to the usual checksum test, since it is acceptable for the received signal to deviate from the transmitted signal.

Note that RawOFDM leverages standard OFDM's use of a preamble and pilot bins separate from data bins [36] to perform the traditional functions of synchronization,

carrier frequency offset (CFO) estimation, channel estimation, and phase tracking.

4.3.2 Whitening

One disadvantage of the simple design described above is that the distribution of the input signal determines the spectral power distribution of the transmitted signal. Any requirements imposed on the distribution put a significant burden on the application designer. Furthermore, the RawOFDM user must pay close attention the range of allowed values to avoid clipping distortion.

RawOFDM reduces this burden by applying *whitening* to the input signal. Whitening is similar to the conventional pseudo-random scrambling and interleaving of the bit-stream, but operates on the real input samples. The aim of whitening is to transform the input signal so that the transformed signal has the distribution of a memory-less Gaussian random variable. RawOFDM takes a packet of signal samples from the application and performs a series of pseudo-random decorrelating transformations: reordering (permutations), sign scrambling and Hadamard matrix multiplication. These transformations are data-independent and orthogonal thus can easily be inverted at the receiver.

The whitening procedure assumes that the packet contains a round number of OFDM symbols. If not, the packet is padded with zero samples. After padding, the input samples can be arranged into an $m \times n$ matrix X of m frequency bins in n OFDM symbols. Then, let us define three operations:

$$\begin{aligned} \mathit{PermuteColumns}(X)[i,j] &= X[i, P_i[j]] \\ \mathit{ScrambleSign}(X)[i,j] &= X[i,j]\ S[i,j] \\ \mathit{Spread}(X) &= HX \end{aligned}$$

where P is a pseudo-random permutation of n items (m such permutations), S is a pseudo-random $m \times n$ matrix of 1 and -1 and H is an $m \times n$ Hadamard matrix. The

whitened signal X^* is computed as:

$$X' = ScrambleSign(PermuteColumns(X)) \tag{4.1}$$

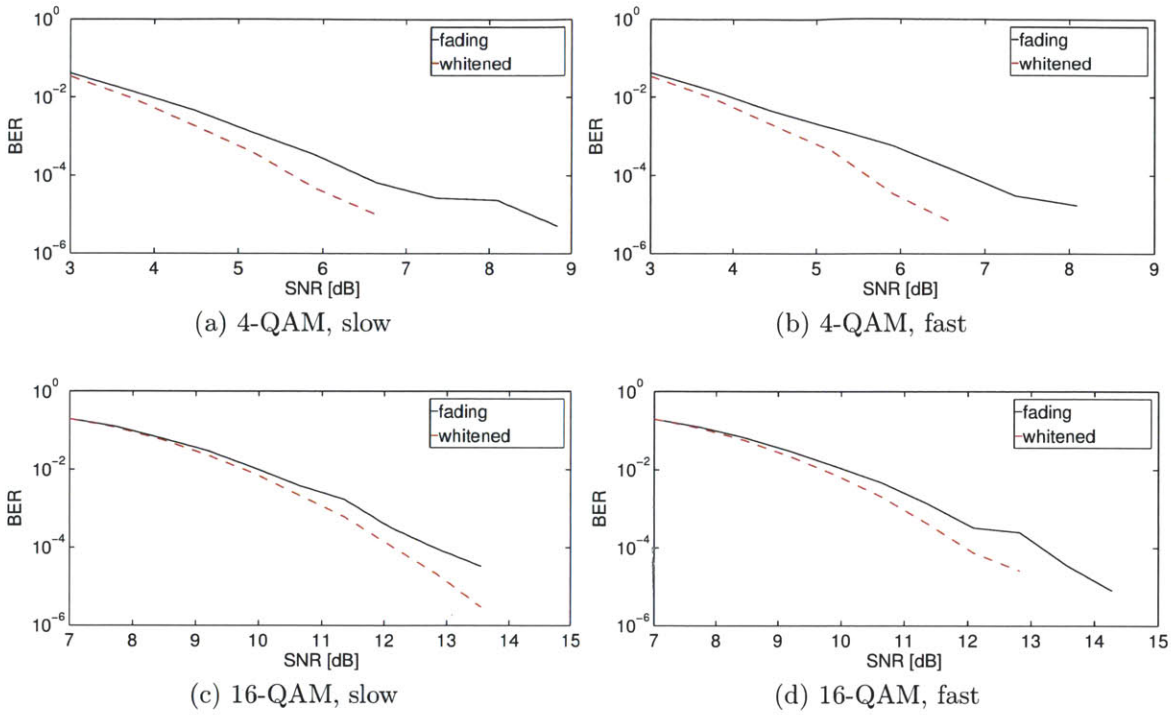
$$X'' = Spread(X') \tag{4.2}$$

$$X^* = ScrambleSign(PermuteColumns(X'')) \tag{4.3}$$

The first step decorrelates the rows of the input signal. This ensures that the second step distributes the energy in the signal evenly between the rows of the matrix, i.e. different frequency bins. This is because all entries in the Hadamard matrix are either 1 or -1. If the covariance matrix of the rows of X is Λ, then the covariance matrix after the multiplication is $H^T \Lambda H$. If X is decorrelated and Λ is diagonal with diagonal entries λ_i, then each element of the diagonal after multiplication is $\sum_i \lambda_i$. With scaling adjusted to m, the Hadamard matrix evenly distributes the energy across the rows (i.e., frequency bins). However, it also introduces correlation between the rows. Hence, the last step decorrelates the rows of the resulting signal. Also by the central limit theorem, the statistical distribution of the output signal approaches Gaussian, as the Hadamard matrix multiplication step performs addition of independent signals.

Whitening brings three benefits to RawOFDM. First, it ensures a flat power spectrum of the baseband signal regardless of the input distribution. This frees the application designer from the concerns regarding the spectrum shape. Second, it reduces the peak-to-average power ratio (PAPR) to that of a Gaussian noise. This is a minor benefit as PAPR can be further reduced using tone reservation or partial tone mapping techniques which qualify as universal, since they do not involve the outer channel coder in the process [108]. Finally, it masks frequency-selective fading and can improve performance beyond simple interleaving. I will explain the latter effect in more detail.

Frequency-selective fading: In a wide-band OFDM system, the gain of individual subcarriers depends on the multi-path propagation through the environment. Thus,



(a) 4-QAM, slow

(b) 4-QAM, fast

(c) 16-QAM, slow

(d) 16-QAM, fast

Figure 4-4: **Masking frequency-selective fading by whitening.** Performance of the convolutional 1/2-rate code (133, 171) interleaved and coupled with 4-QAM and 16-QAM as in 802.11a/g bit-rates 12Mb/s and 24 Mb/s (top to bottom) over a channel with simulated frequency fading (slow on left, fast on right). Whitening consistently improves performance in all tested scenarios.

depending on the environment, two bins which frequencies are farther apart than the coherence frequency of the channel will observe independent levels of attenuation. After equalization, the gain is normalized, but the noise variance is no longer uniform. (See Section 2.2.3 for more background on the topic.)

Figure 4-4 shows the performance of the digital code in 802.11a/g standard comprised of the convolutional 1/2-rate code with polynomial (133, 171) interleaved and coupled with 4-QAM and 16-QAM. In this experiment, 1Mbit of data is encoded into 200-symbol packets, where each OFDM symbol carries data in 48 complex subcarriers. Each packet is subjected to Rayleigh frequency-selective fading, and the resulting bit-error rate (BER) is recorded. The figure presents the measured BER for the overall channel SNR. In the figure, the series "fading" shows to the performance using frequency interleaving only, while "whitened" shows to the performance using

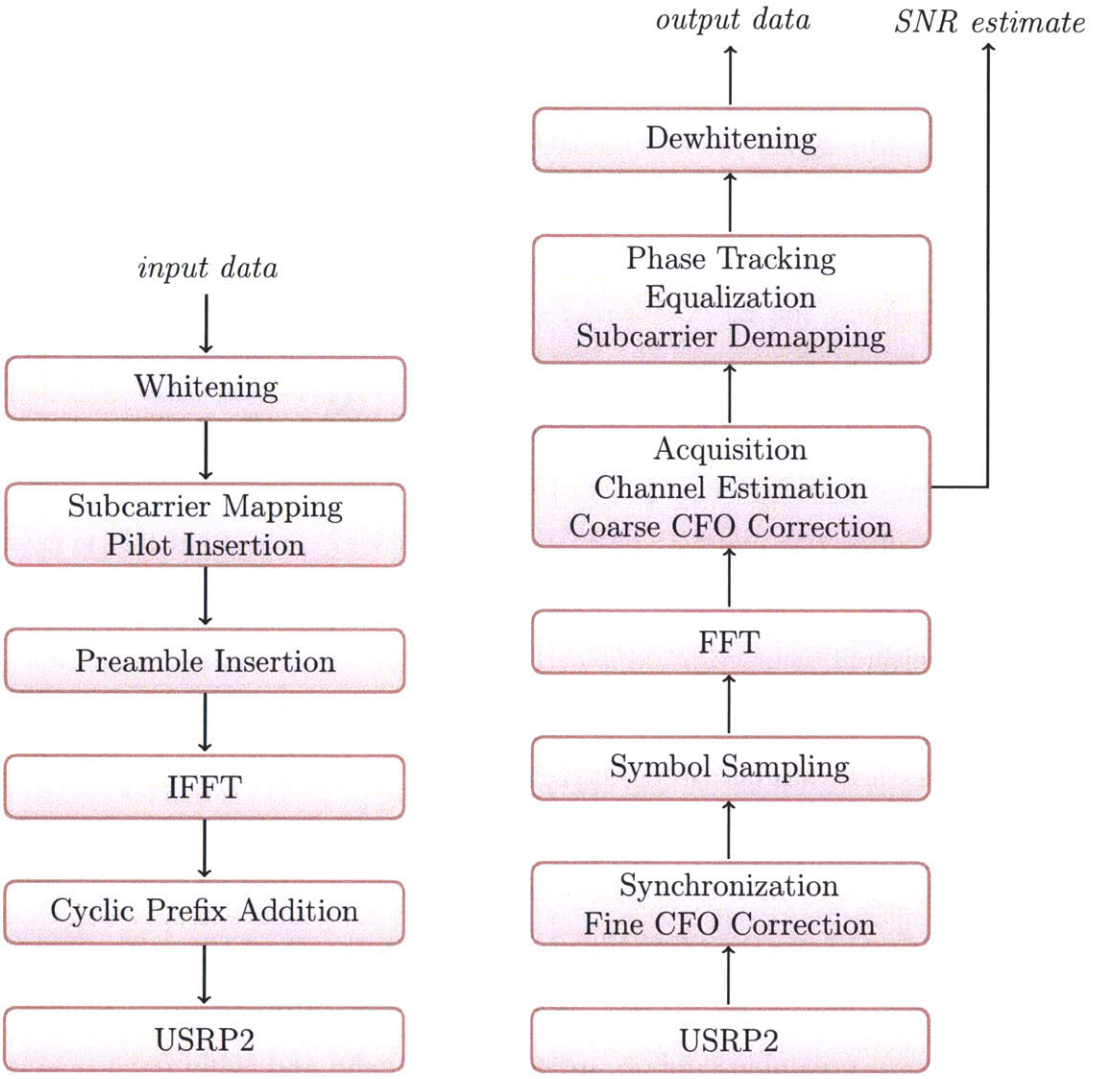
RawOFDM whitening. Whitening consistently improves performance in all tested scenarios. For set BER of 10^{-4} the improvement ranged up to 2dB.

4.3.3 Software Radio Implementation

RawOFDM is implemented in the GNURadio software radio framework [33]. At the time of writing, GNURadio includes an OFDM implementation contributed by the project leadership. However, that implementation uses non-universal phase tracking and equalization. In RawOFDM, phase tracking uses pilot tones only and thus is independent of the outer channel code.

Figure 4-5 shows the block diagram of the implementation. Each block corresponds to a GNURadio signal processing block, although the Synchronization block is hierarchically composed of more basic blocks. Note slight differences from the diagram of an abstract OFDM in Fig. 2-8. Here preambles are defined in the frequency domain. The subcarrier mapping and demapping blocks correspond to the concept shown in Fig. 4-2.

At the receiver, synchronization is performed using the Schmidl-Cox method [83] which uses the pseudo-random preamble symbol to detect the beginning of a frame and also yields a fine estimate of the carrier frequency offset (in the range $[-f_c/2, f_c/2)$ if f_c is the subcarrier frequency width). Symbol sampler removes the cyclic prefix by sampling an OFDM symbol at a time offset indicated by the synchronizer. The *Acquisition* block correlates the preamble to determine and correct the coarse frequency offset (an integer multiply of f_c) but also to reject false positives from the synchronizer. The known preamble symbols are used to estimate and equalize the channel.



(a) Transmitter (b) Receiver

Figure 4-5: **RawOFDM implementation.** Block diagram of the GNURadio implementation of RawOFDM.

Chapter 5

Performance Evaluation and Analysis

In this chapter, I present results of an experimental evaluation of SoftCast in a physical testbed as well as information-theoretic analysis of SoftCast's linear code.

5.1 Evaluation Environment

I use the GNURadio codebase [33] to build a prototype of SoftCast and an evaluation infrastructure to compare it against two baselines:

- MPEG4 (i.e., H.264) over an 802.11 PHY.
- Layered video where the video is coded using the scalable video extension (SVC) of H.264/AVC [50] and is transmitted over hierarchical modulation [23]. This approach has been proposed in [49] to extend Digital TV to mobile handheld devices.

The Physical Layer. Since both baselines and SoftCast use OFDM, I built RawOFDM, which can be used as a shared physical layer that allows the execution to branch depending on the evaluated video scheme. For MPEG and SVC, I also developed software modules that perform 802.11 interleaving, convolutional coding,

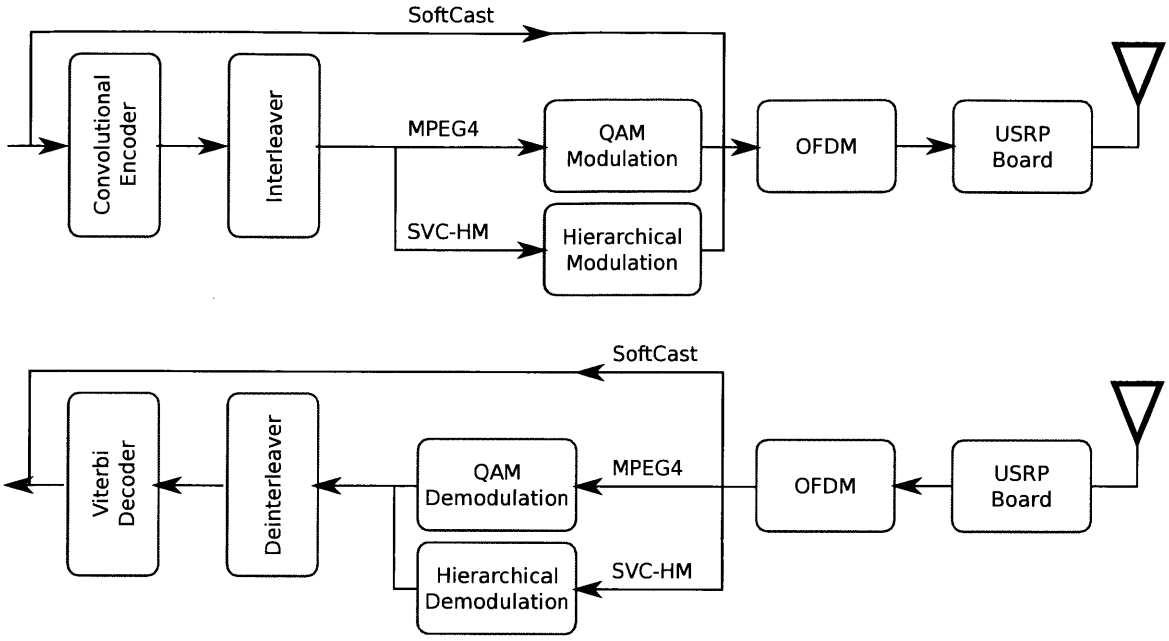


Figure 5-1: **Block diagram of the evaluation system.** The top graph shows the transmitter side the bottom graph shows the receiver.

and Viterbi decoding using the Spiral project [90]. When combined with those digital channel codes, RawOFDM is equivalent to the modern PHY as embodied in 802.11a/g, and similar (modulo specific parameters) to 802.11n, 802.16 or DVB-T/T2/H.

Fig. 5-1 shows a block diagram of the implemented PHY layer. On the transmit side, the PHY passes SoftCast's packets directly to OFDM, whereas MPEG4 and SVC-encoded packets are subject to convolutional coding and interleaving, where the code rate depends on the chosen bit rate. MPEG4 packets are then passed to the QAM modulator while SVC-HM packets are passed to the hierarchical modulation module. The last step involves OFDM transmission using RawOFDM and is common to all schemes. On the receive side, the signal is passed to the RawOFDM module which performs carrier frequency offset (CFO) estimation and correction, channel estimation and correction, and phase tracking. The receiver then inverts the execution branches at the transmitter.

Video Coding. I implemented SoftCast in Python (with SciPy). For the baselines, I used reference implementation available online. Specifically, I generate MPEG-4

streams using the H.264/AVC [41,78] codec provided in open source FFmpeg software and the x264 codec library [24,103]. We generate the SVC stream using the JSVM implementation [50], which allows us to control the number of layers. Also for MPEG4 and SVC-HM, I add an outer Reed-Solomon code for error protection with the same parameters as used for digital TV [23]. All the schemes: MPEG4, SVC-HM, and SoftCast use a GoP of 16 frames.

Testbed: We run our experiments in the 20-node GNURadio testbed shown in Fig. 5-2. Each node is a laptop connected to a USRP2 radio board [94]. We use the RFX2400 daughterboards which operate in the 2.4 GHz range.

Modulation. The conventional design represented by MPEG4 over 802.11 uses the standard modulation and FEC, i.e., BPSK, QPSK, 16QAM, 64QAM and 1/2, 2/3, and 3/4 FEC code rates. The hierarchical modulation scheme uses QPSK for the base layer and 16QAM for the enhancement layer as recommended in [56]. It is allowed to control how to divide transmission power between the layers to achieve the best performance [56]. The three layer video uses QPSK at each level of the QAM hierarchy and also controls power allocation between layers. SoftCast is transmitted directly over OFDM. The OFDM parameters are selected to match those of 802.11a/g.

The Wireless Environment. The carrier frequency is 2.4 GHz which is the same as that of 802.11b/g. The channel bandwidth after decimation is 1.25 MHz. Since the USRP radios operate in the same frequency band as 802.11 WLANs, there is unavoidable interference. To limit the impact of interference, I run the experiments at night. Each experiment is repeated five times and interleave runs of the three compared schemes.

Metric: The schemes are compared using the Peak Signal-to-Noise Ratio (PSNR). It is a standard metric for video quality [79] and is defined as a function of the mean squared error (MSE) between all pixels of the decoded video and the original as

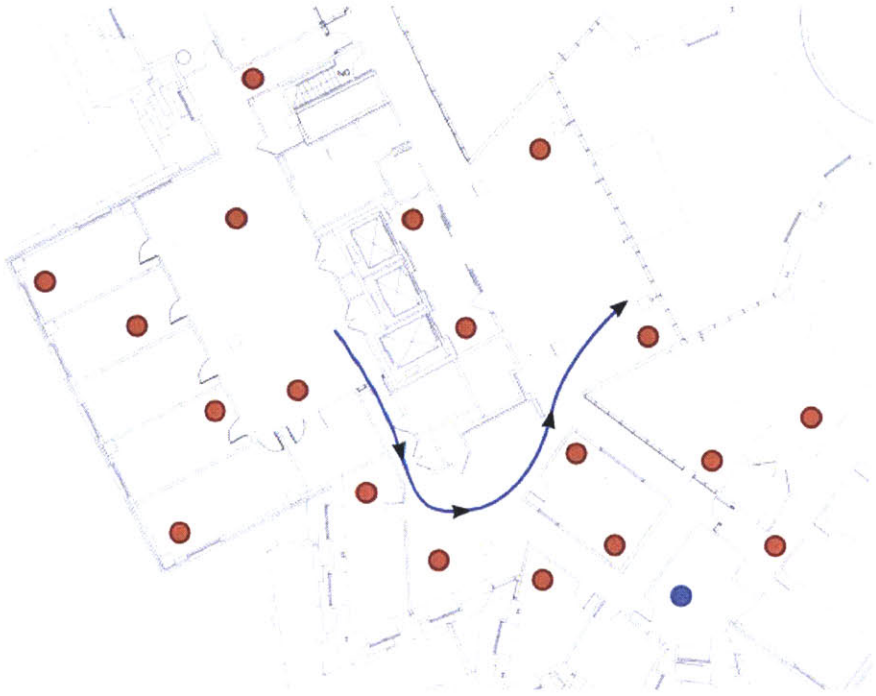


Figure 5-2: **Testbed.** Dots refer to nodes; the line shows the path of the receiver in the mobility experiment when the blue dot was the transmitter.

follows:

$$PSNR = 10 \log_{10} \frac{2^L - 1}{MSE} \quad [dB],$$

where L is the number of bits used to encode pixel luminance, typically 8 bits. A PSNR below 20 dB refers to *bad* video quality, and differences of 1 dB or higher are visible [79].

Test Videos: We use standard reference videos in the SIF format (352×240 pixels, 30 fps) from the Xiph [105] collection. Since codec performance varies from one video to another, we create one monochrome 480-frame test video by splicing 1 second from each of 16 popular reference videos: *akiyo, bus, coastguard, crew, flower, football, foreman, harbour, husky, ice, news, soccer, stefan, tempete, tennis, waterfall.*

Other Parameters: The packet length is 14 OFDM symbols or 250 bytes when using 16QAM with 1/2 FEC rate. The transmission power is 100mW. The channel bandwidth is 1.25 MHz. Note that all experiments in this work use the same transmission power and the same channel bandwidth. Thus, the compared schemes

are given the same channel capacity[1] and differences in their throughput and their streaming quality are due only to how effectively they use that capacity.

5.2 Results

I empirically evaluate SoftCast and compare it against: 1) the conventional design, which uses MPEG4 over 802.11 and 2) SVC-HM, a state of the art layered video design that employs the scalable video extension of H.264 and a hierarchical modulation PHY layer [56,85].

5.2.1 Benchmark Results

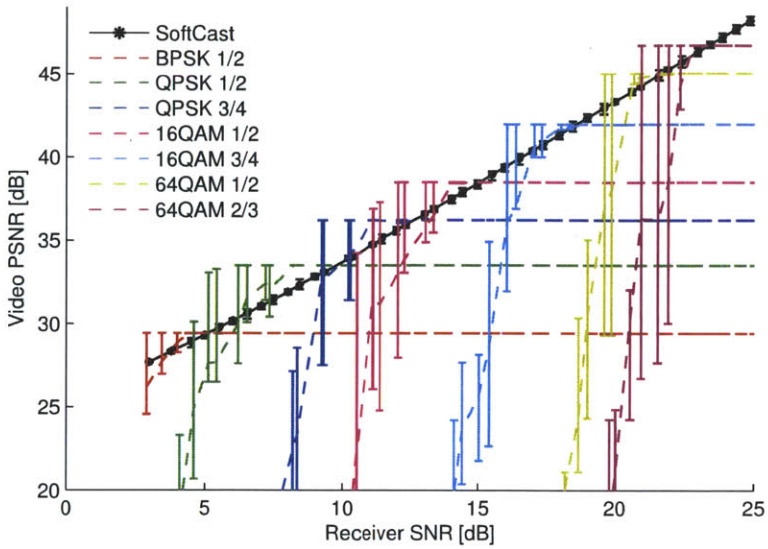
Let us first revisit the result in §1.2.1, which is reproduced in Fig. 5-3 for convenience.

Method: In this experiment, I pick a node randomly in the testbed, and make it broadcast the video using the conventional design, SoftCast, and SVC-HM. I run MPEG4 over 802.11 for all 802.11 choices of modulation and FEC code rates. I also run SVC-HM for the case of 2-layer and 3-layer video. During the video broadcast, all nodes other than the sender act as receivers. [2] For each receiver, I compute the average SNR of its channel and the PSNR of its received video. To plot the video PSNR as a function of channel SNR, I divide the SNR range into bins of 0.5 dB each, and take the average PSNR across all receivers whose channel SNR falls in the same bin. This produces one point in Fig. 5-3. I use this procedure to produce points for all lines in the figure. I repeat the experiment by randomly picking the video source from the nodes in the testbed.

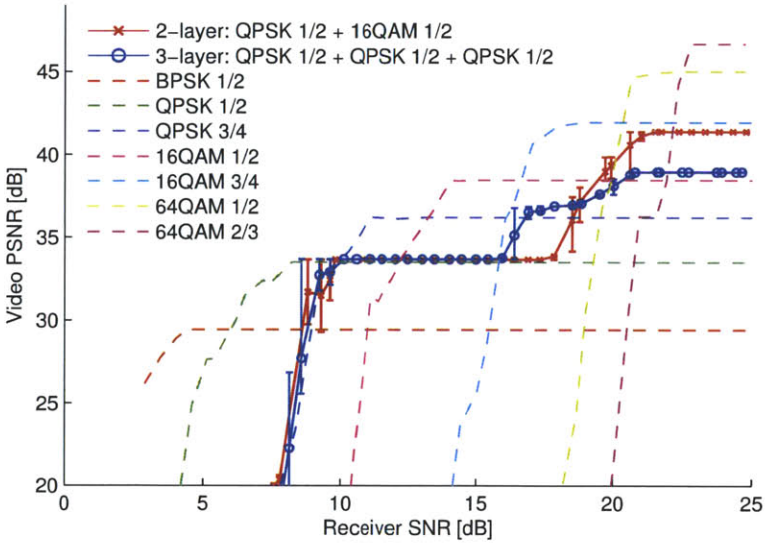
Results: Fig. 5-3 shows that for any choice of 802.11 modulation and FEC code rate, there exists a critical SNR below which the conventional design degrades sharply,

[1] Shannon capacity is $C = W\log(1 + \frac{HP}{WN})$ where W is the bandwidth, P is the power, H is the channel function, and N is the noise power per Hz.

[2] I decode the received video packets offline because the GNUradio Viterbi decoder can not keep up with packet reception rate.



(a) SoftCast vs. Conventional Design



(b) SVC-HM vs. Conventional Design

Figure 5-3: **Basic benchmark.** The figure shows average video quality as a function of channel quality. The bars show differences between the maximum and minimum quality, which are large around cliff points. The top graph compares SoftCast (black line) against the conventional design of MPEG4 over 802.11 (dashed lines) for different choices of 802.11 modulation and FEC code rate. The bottom graph compares layered video (red and blue lines) against the conventional design.

and above it the video quality does not improve with channel quality. In contrast, SoftCast's PSNR scales smoothly with the channel SNR. Further, SoftCast's PSNR matches the envelope of the conventional design curves at each SNR. The combina-

tion of these two observations means that SoftCast can significantly improve video performance for mobile and multicast receivers while maintaining the efficiency of the existing design for the case of a single static receiver.

It is worth noting that this does not imply that SoftCast outperforms MPEG4. MPEG4 is a compression scheme that compresses video effectively, whereas SoftCast is a wireless video transmission architecture. The inefficacy of the MPEG4-over-802.11 lines in Fig. 5-3a stems from the fact that the conventional design separates video coding from channel coding. The video codec (MPEG and its variants) assumes an error-free lossless channel with a specific transmission bit rate, and given these assumptions, it effectively compresses the video. However, the problem is that in scenarios with multiple or mobile receivers, the wireless PHY cannot present an error-free lossless channel to all receivers and at all times without reducing everyone to a conservative choice of modulation and FEC and hence a low bit rate and a corresponding low video quality.

Fig. 5-3b shows that a layered approach based on SVC-HM exhibits milder cliffs than the conventional design and can provide quality differentiation. However, layering reduces the overall performance in comparison with conventional single layer MPEG4. Layering incurs overhead both at the PHY and the video codec. At any fixed PSNR in Fig. 5-3b, layered video needs a higher SNR than the single layer approach to achieve the same PSNR. This is because in hierarchical modulation, every higher layer is noise for the lower layers. Similarly, at any fixed SNR, the quality of the layered video is lower than the quality of the single layer video at that SNR. This is because layering imposes additional constraints on the codec and reduces its compression efficiency [100].

5.2.2 Multicast

Method. We pick a single sender and three multicast receivers from the set of nodes in our testbed. The receivers' SNRs are 11 dB, 17 dB, and 22 dB. In the conventional design, the source uses the modulation scheme and FEC that correspond to 12 Mb/s 802.11 bit rate (i.e., QPSK with 1/2 FEC code rate) as this is the highest bit rate

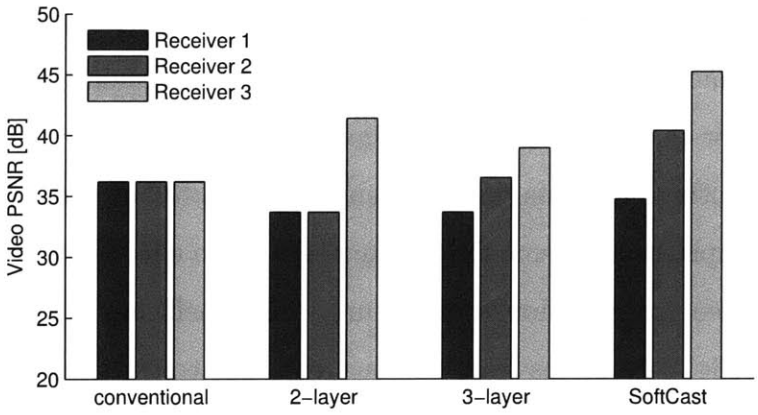


Figure 5-4: **Multicast to three receivers.** The figure shows that layering provides service differentiation between receivers as opposed to single layer MPEG4. But layering incurs overhead at the PHY and the codec, and hence extra layers reduce the maximum achievable video quality. In contrast, SoftCast provides service differentiation while achieving a higher overall video quality.

supported by all three multicast receivers. In 2-layer SVC-HM, the source transmits the base layer using QPSK and the enhancement layer using 16 QAM, and protects both with a half rate FEC code. In 3-layer SVC-HM, the source transmits each layer using QPSK, and uses a half rate FEC code.

Results: Fig. 5-4 shows the PSNR of the three multicast receivers. The figure shows that, in the conventional design, the video PSNR for all receivers is limited by the receiver with the worse channel. In contrast, 2-layer and 3-layer SVC-HM provide different performance to the receivers. However, layered video has to make a trade-off: The more the layers the more performance differentiation but the higher the overhead and the worse the overall video PSNR. SoftCast does not incur a layering overhead and hence can provide each receiver with a video quality that scales with its channel quality, while maintaining a higher overall PSNR.

Method: Next, let us focus on how the diversity of channel SNR in a multicast group affects video quality. I create 40 different multicast groups by picking a random sender and different subsets of receivers in the testbed. Each multicast group is parametrized by its SNR span, i.e., the range of its receivers' SNRs. I keep the average SNR of all multicast groups at 15 ($\pm$1) dB. I vary the range of the SNRs in

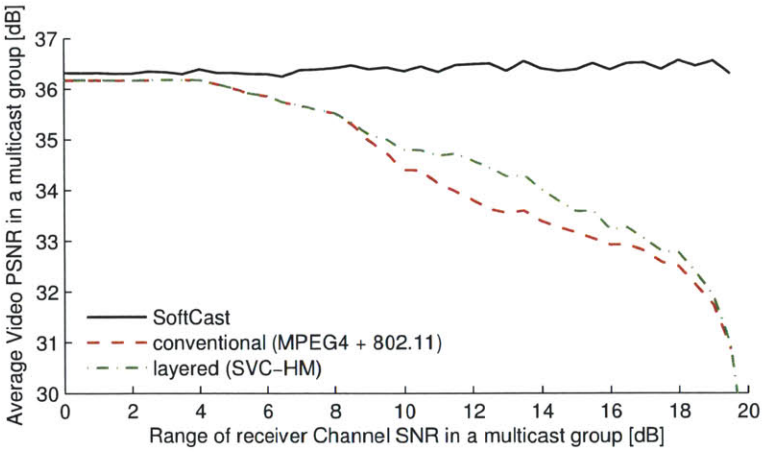


Figure 5-5: **Serving a multicast group with diverse receivers.** The figure plots the average PSNR across receivers in a multicast group as a function of the SNR range in the group. The figure shows that the conventional design and SVC-HM provide a significantly lower average video quality than SoftCast for multicast group with a large SNR span.

the group from 0-20 dB by picking the nodes in the multicast group. Each multicast group has up to 15 receivers, with multicast groups with zero SNR range having only one receiver. For each group, I run each of the three compared schemes. The transmission parameters for each scheme (i.e., modulation and FEC rate) is such that provides the highest bit rate and average video quality without starving any receiver in the group. Finally, SVC-HM is allowed to pick for each group whether to use one layer, two layers, or three layers.

Results. Fig. 5-5 plots the average PSNR in a multicast group as a function of the range of its receiver SNRs. It shows that SoftCast delivers a PSNR gain of up to 5.5 dB over both the conventional design and SVC-HM. One may be surprised that the PSNR improvement from layering is small. Looking back, Fig. 5-4b shows that layered video does not necessarily improve the average PSNR in a multicast group. Rather, it changes the set of realizable PSNRs from the case of a single layer where all receivers obtain the same PSNR to a more diverse PSNR set, where receivers with better channels can obtain higher video PSNRs.

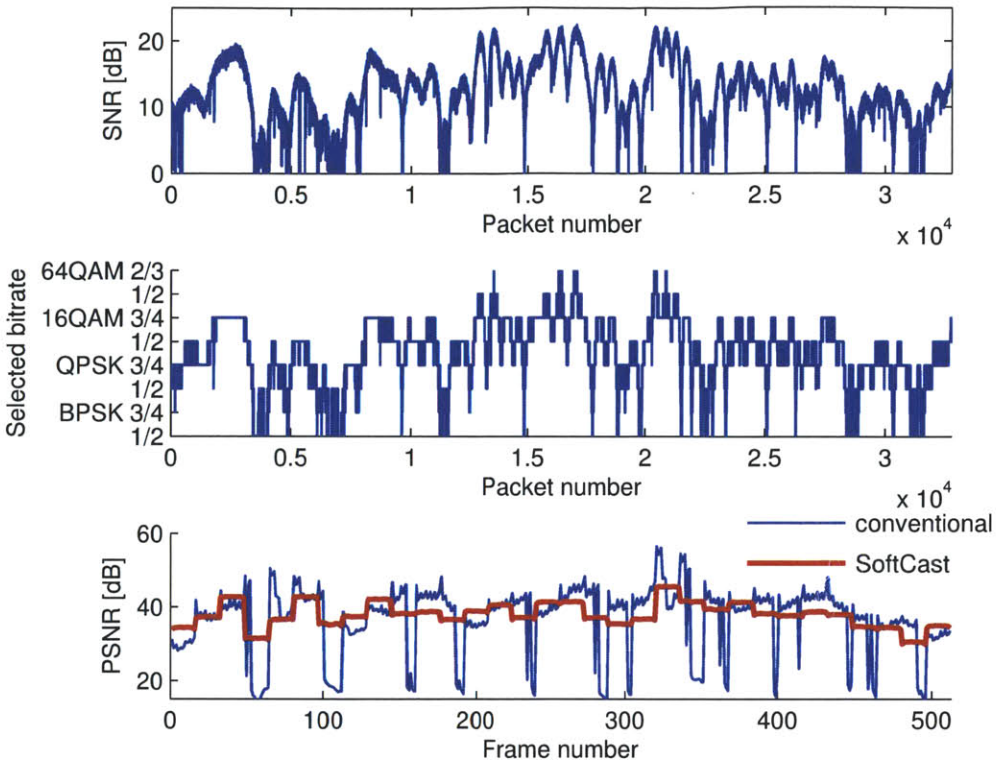


Figure 5-6: **Performance under mobility.** The figure compares the video quality of the conventional design and SoftCast under mobility. The conventional design is allowed to adapt its bitrate and video code rate. The top graph shows the SNR of the received packets, the middle graph shows the transmission bit rate chosen by SoftRate and used in the conventional design. The bottom graph plots the per frame PSNR. The figure shows that even with rate adaptation, a mobile receiver still suffers significant glitches with the conventional design. In contrast, SoftCast can eliminate these glitches.

5.2.3 Mobility

Next, I study video glitches experienced by a single mobile receiver. Since a video PSNR below 20 dB is not watchable [65], I identify glitches as frames whose PSNR is below 20 dB.

Method: Performance under mobility is sensitive to the exact movement patterns. Since it is not possible to repeat the exact movements across experiments with different schemes, I follow a trace-driven approach like the one used in [98]. Specifically, I perform the mobility experiment with non-video packets. I then subtract the received soft values from the transmitted soft values to extract the noise pattern on the chan-

nel. This noise pattern contains all necessary information to describe the distortion that occurred on the channel including fading, interference, the effect of movement, etc. I then apply the same noise pattern to each of the three video transmission schemes to emulate its transmission on the channel. This allows us to compare the performance of the three schemes under the same conditions. Fig. 5-2 shows the path followed during the mobility experiments.

The conventional design is allowed to adapt its transmission bit rate and video code rate. To adapt the bit rate it uses SoftRate [98], which is particularly designed for mobile channels. To adapt the video code rate, MPEG4 is allowed to switch the video coding rate at GoP boundaries to match the transmission bit rate used by SoftRate. Adapting the video faster than every GoP is difficult because frames in a GoP are coded with respect to each other. The conventional design is also allowed to retransmit lost packets with the maximum retransmission count set to 11. This scheme does not adapt the bit rate or video code rate of layered video. This is because a layered approach should naturally work without adaptation. Specifically, when the channel is bad, the hierarchical modulation at the PHY should still decode the lower layer, and the video codec should also continue to decode the base layer. Finally, SoftCast is not allowed to adapt its bit rate or its video code rate nor is it allowed to retransmit lost packets.

Results: Fig. 5-6 shows the results of our experiment. The top graph shows the SNR in the individual packets in the mobility trace. Fig 5-6b shows the transmission bit rates picked by SoftRate and used in the conventional design. Fig 5-6c shows the per-frame PSNR for the conventional design and SoftCast. The results for SVC-HM are not plotted because SVC-HM failed to decode almost all frames (80% of GoP were not decodable). This is because layering alone, and particularly hierarchical modulation at the PHY, could not handle the high variability of the mobile channel. Recall that in hierarchical modulation, the enhancement layers are effectively noise during the decoding of the base layer, making the base layer highly fragile to SNR dips. As a result, the PHY is not able to protect the base layer from losses. In

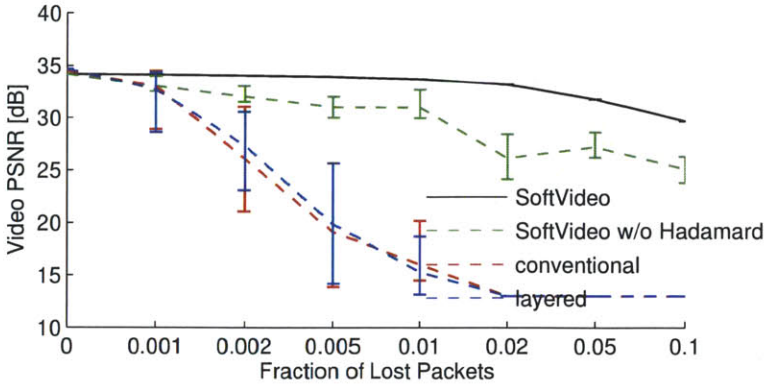


Figure 5-7: **Resilience to packet loss.** The figure shows that both SVC-HM and the conventional MPEG-based design suffer dramatically at a packet loss rate as low as 0.5%. In contrast, SoftCast's is only mildly affected even when the loss rate is as high as 10%. For reference, the figure shows the performance of SoftCast if it did not use the Hadamard matrix to ensure that all packets carry equal amount of information.

contrast single layer video reacted better to SNR variability because its PHY can adapt to use BPSK which is the most robust among the various modulation schemes.

Fig 5-6c shows that, with mobility, the conventional wireless design based on MPEG-4 experiences significant glitches in video quality. These glitches happen when a drop in the transmission bit rate causes significant packet losses such that even if the packets are recovered with retransmissions, they might still prevent timely decoding of the video frames. In comparison, SoftCast's performance is stable even in the presence of mobility. This is mainly due to SoftCast being highly robust to packet loss due to that it avoids Huffman and differential encoding and it spreads the video information across all packets. The results in Fig 5-6c show that, in this mobile experiment, 14% of the frames transmitted using the conventional design suffer from glitches. SoftCast however has eliminated all such glitches.

5.2.4 Resilience to Packet Loss

Method: I pick a random pair of nodes from the testbed and transmit video between them. I generate packet loss by making an interferer transmit at constant intervals. By controlling the interferer's transmission rate I can control the packet

loss rate. I compare four schemes: the conventional design based on MPEG4, 2-layer SVC-HM, full-fledged SoftCast, and SoftCast after disabling the Hadamard multiplication. I repeat the experiment for different transmission rates of the interferer.

Results: Fig. 5-7 reports the video PSNR at the receiver across all compared schemes as a function of the packet loss rate. The figure has a log scale. It shows that in both baselines the quality of video drops sharply even when the packet loss rate is less than 0.5%. This is because both the MPEG4 and SVC codecs introduce dependencies between packets due to Huffman encoding, differential encoding and motion compensation, as a result of which the loss of a single packet within a GoP can render the entire GoP undecodable. In contrast, SoftCast's performance degrades only gradually as packet loss increases, and is only mildly affected even at a loss rate as high as 10%. The figure also shows that Hadamard multiplication significantly improves SoftCast's resilience to packet loss. Interestingly, SoftCast is more resilient than MPEG4 even in the absence of Hadamard multiplication.

SoftCast's resilience to packet loss comes from:

- The use of a 3D DCT ensures that all SoftCast packets include information about all pixels in a GoP, hence the loss of a single packet does not create patches in a frame, but rather distributes errors smoothly across the entire GoP.

- SoftCast packets are not coded relative to each other as is the case for differential encoding or motion compensation. Hence the loss of one packet does not prevent the decoding of other received packets.

- All SoftCast packets have equal energy as a result of Hadamard multiplication, and hence the decoding quality degrades gracefully as packet losses increase. The LLSE decoder, in particular, leverages this property to decode the GoP even in the presence of packet loss.

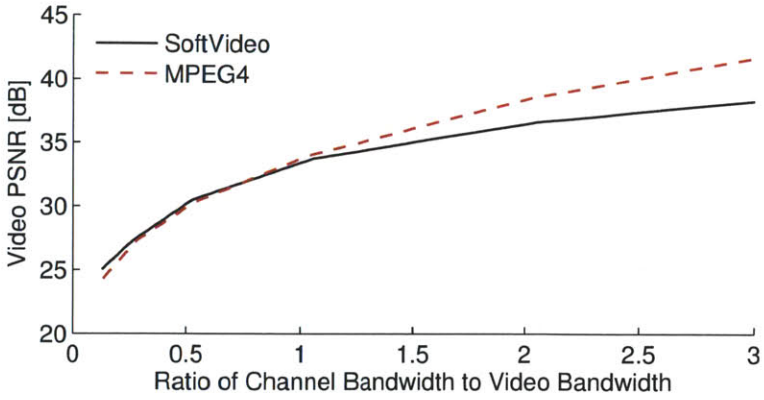


Figure 5-8: **Impact of available wireless bandwidth.** The figure plots the performance of SoftCast and MPEG4 for a single receiver with 10 dB SNR as a function of the ratio of wireless bandwidth to video bandwidth (i.e., pixels/s). SoftCast is suitable for environments where it is desirable to send a large video over a relatively low bandwidth channel.

5.2.5 Impact of Available Wireless Bandwidth

Next, let us explore SoftCast's limitations. SoftCast is designed for environments where the wireless bandwidth is the bottleneck, i.e., the video source is too big to fit within the available channel bandwidth. (Note, if a 20MHz channel is shared by 10 users, then the available bandwidth per user is 2MHz.) The source bandwidth is typically defined as the number of dimensions/sec, which in the case of a video source refers to the number of pixel values per second [16]. If the available wireless bandwidth is less than the video source bandwidth, SoftCast compresses the video by dropping low energy 3D DCT frequencies. However, SoftCast's existing design has no particular approach to deal with environments where the source's bandwidth may be higher than the wireless bandwidth. The conventional design can leverage such scenarios to make a wideband low SNR channel perform as if it were a high SNR narrow bandwidth channel, using an approach called bandwidth expansion [16,88]. However, I am unaware of good linear codes for bandwidth expansion. A straightforward linear code would simply repeat the same signal; however repetition is not efficient. Below, I show empirical results from scenarios with bandwidth expansion.

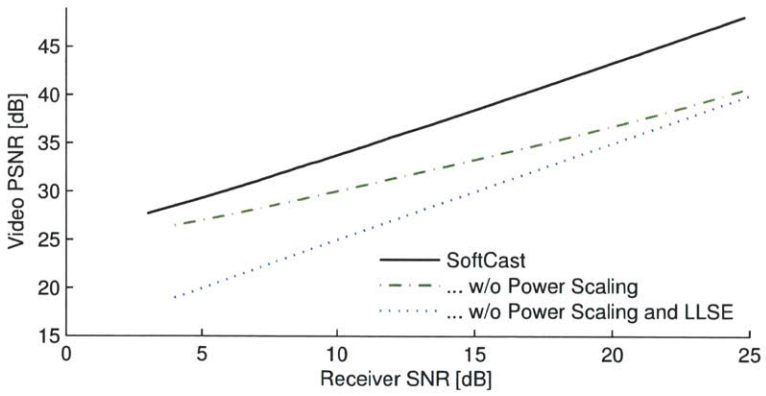


Figure 5-9: **SoftCast microbenchmark** The figure plots the contributions of SoftCast's components to its video quality. The figure shows that the use of LLSE is particularly important at low SNRs where as error protection via power scaling is important at high SNRs.

Method: I pick a single sender-receiver pair with 10 dB SNR. I vary the available wireless bandwidth by changing the packet rate on the USRP2, and transmit the same video with both with SoftCast and MPEG4. For scenarios that require bandwidth expansion SoftCast simply repeats the signal. As for MPEG4, the 802.11-like PHY naturally performs bandwidth expansion.

Results: Fig. 5-8 shows that SoftCast remains competitive in a wide range of scenarios where the wireless bandwidth is smaller than the source bandwidth. In scenarios where wireless bandwidth is significantly larger, SoftCast is unable to efficiently utilize the bandwidth. This is a limitation of SoftCast's linear design which given surplus bandwidth can only apply repetition coding. However, the wireless bandwidth is a shared scarce resource. Hence, I believe, most practical scenarios are limited by the wireless bandwidth, and can benefit from SoftCast's design.

5.2.6 Microbenchmark

I examine the contribution of SoftCast's components to its performance.

Method: I pick a sender receiver pair at random. I vary the SNR by varying the transmission power at the sender. For each SNR, I make the sender transmit the

video with SoftCast, SoftCast with linear scaling disabled, and SoftCast with both linear scaling and LLSE disabled. I repeat the experiments multiple times and report the average performance for each SNR value.

Results: The figure shows that SoftCast's approach to error protection based on linear scaling and LLSE decoding contributes significantly to its resilience. Specifically, linear scaling is important at high SNRs since it amplifies fine image details and protects them from being lost to noise. In contrast, the LLSE decoder is important at low SNRs when receiver measurements are noisy and cannot be trusted, because it allows the decoder to leverage its knowledge of the statistics of the DCT components.

5.3 Theoretical Analysis

The results of the experimental evaluation indicate that SoftCast not only provides desired graceful degradation. but also remains competitive with single-layer video on stationary channels while outperforming multi-layer video on broadcast channels. In order to provide more insight into these results, in this section, I analyze the performance of the linear code employed by SoftCast and compare it to the theoretical upper bound on the performance of the separate codec-PHY design. However, to do so analytically, I simplify the model of the source signal to that of a correlated multivariate Gaussian. The channel is modeled as multidimensional with additive white Gaussian noise (AWGN). Throughout this section, I refer to the linear code as **analog**, while the separate design is correspondingly named **digital**.

5.3.1 Preliminaries

I present the problem model and review basic results for communications, both digital and analog, of Gaussian sources over Gaussian channels. For this section, I limit the exposition to the major concepts rather than the formal definition and proofs. Details and excellent textbook treatment for the digital results can be found in [16].

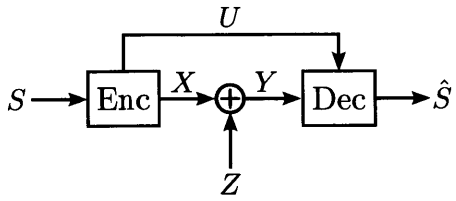


Figure 5-10: **A system model for communications.** The input $S \in \mathbb{R}^N$ is encoded and then transmitted through M parallel AWGN channels. The decoder outputs a reproduction $\hat{S}$ with performance determined by the MSE. A side channel may exist between the encoder and decoder to transmit a small amount of metadata to aid signal reconstruction.

Problem Statement

Consider the system depicted in Figure 5-10. At each time step, the encoder accepts an input vector $S \in \mathbb{R}^N$ and transmits an output vector $X \in \mathbb{R}^M$over a set of parallel channels perturbed by an additive white Gaussian noise (AWGN) vector Z. The decoder receives the noisy transmission $Y = X + Z$ and produces a reproduction $\hat{S}$ of the transmitted signal. The goal of the system is to minimize the distortion between S and $\hat{S}$, measured as the mean-squared error (MSE), i.e., $D = E\left[\sum_i (\hat{S}_i - S_i)^2\right]$. The constraint is on the the average power of the encoder, meaning

$$E\left[\frac{1}{M}\sum_i X_i^2\right] \leq P.$$

For this chapter, we assume that the source S is the class of memoryless multivariate Gaussian vectors with diagonal covariance matrices. Hence, each component or *subband* S_i of the source is an independent zero-mean Gaussian variable with variance λ_i. Assume a natural order of subbands according to variances such that $\lambda_1 \geq \cdots \geq \lambda_N$. Moreover, assume the noise elements Z_i are iid with variance σ^2. The channel can also be characterized using a signal-to-noise ratio $\text{SNR} = P/\sigma^2$.

Section 5.3.3 discusses a two-user broadcast channel, which is shown in Figure 5-11. For this system, each user may have different channel statistics. We define the noise variances as σ_1^2 and σ_2^2 (with corresponding SNR_1 and SNR_2) and assume $\sigma_1 \leq \sigma_2$.

Although practical sources are often correlated both in time and across subbands,

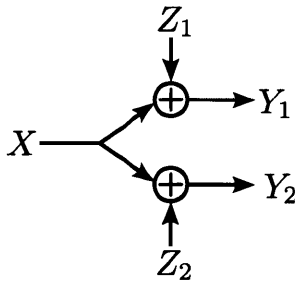


Figure 5-11: **The two-user broadcast channel model.** The transmission X is desired by both users but is corrupted by noise.

an added step of decorrelation or subband decomposition can prepare the source for such a system setup. For example, transform coding (DCT) is the typical first step of video communications [41,99]. Similarly, techniques in channel coding can effectively hide the correlation in the channel behind the AWGN model. For example, OFDM systems employ FFT, guard intervals and pilots to mask frequency fading, ISI and frequency offset [36].

The average power constraint can easily be changed to a per-component power constraint

$$E\left[X_i^2\right] \leq P \quad \forall i$$

by applying a Hadamard matrix to X before transmitting. The Hadamard matrix is an orthogonal transform with entries $+1$ and -1 only that is commonly used to to redistribute energy in a signal [84].

Digital Communications

The separation theorem dictates that the optimal solution for the point-to-point source-channel problem can be obtained by first applying an optimal source coding and then using an optimal channel code to transmit this bit-stream. The capacity $C(\text{SNR})$ of the channel defines the maximum rate of the bit-stream that can be decoded without error for a given channel SNR. Meanwhile, the rate-distortion function of the source, $R(D)$ defines the minimum achievable rate R that yields distortion D. Hence $R(D_{\text{opt}}) = C(\text{SNR})$. This is also known as OPTA (Optimum Performance Theoretically Attainable). Note that achieving an end-to-end distortion of D_{opt} could

require infinite block length in both source and channel coders.

For a set of M AWGN channels with a particular SNR, the capacity is

$$C(\text{SNR}) = \frac{M}{2}\log(1+\text{SNR}). \tag{5.1}$$

For a multi-variate Gaussian S, the rate-distortion function is given by the reverse water-filling solution:

$$R(D) = \sum_{i=1}^{N} \frac{1}{2}\log_2 \frac{\lambda_i}{D_i} \tag{5.2}$$

where $D_i = \min(\gamma, \lambda_i)$ and γ is chosen to satisfy $\sum_i D_i = D$.

By reverse waterfilling, some source subbands may not be transmitted when the channel is noisy. As the channel gets worse, the best choice of γ increases and leads to more subbands being suppressed. For a given source S and channel SNR, we can find the optimal choice of γ and subsequently K, the number of subbands transmitted. The resulting optimal end-to-end distortion is

$$D_{\text{dig}}(\text{SNR}) = K\left(\frac{\prod_{i=1}^{K}\lambda_i}{(1+\text{SNR})^M}\right)^{1/K} + \sum_{i=K+1}^{N}\lambda_i. \tag{5.3}$$

We also survey results related to the Gaussian broadcast channel model of Figure 5-11, where the capacity is known since the channel is *stochastically degraded*, i.e., can be modeled as $Y_2 = X + Z_2 = Y_1 + Z_2'$, where $\sigma_2'^2 = \sigma_2^2 - \sigma_1^2$ [17]. The achievable rate region can be obtained by *superposition coding*, where the encoder splits the bit-stream into low-priority enhancement at rate R_1 and high-priority base at rate R_2, channel-codes them, and adds them up before transmitting. Both users can decode the base and the stronger receiver can also decode the enhancement. The achievable rate region is given by [16]

$$R_1 = \frac{M}{2}\log_2(1+\omega\text{SNR}_1), \tag{5.4}$$

$$R_2 = \frac{M}{2}\log_2\left(1+\frac{(1-\omega)\text{SNR}_2}{\omega\text{SNR}_2+1}\right), \tag{5.5}$$

where $\omega \in [0,1]$ is a parameter that determines the fraction of total power devoted to transmitting the enhancement. Superposition coding in QAM is also known as *hierarchical modulation.*

The matching source coder needs to encode the source into two bit-streams. For example, the high-priority base is result of a coarse quantization, and the low-priority enhancement is a finer quantization of the residual error from the coarse quantization. Gaussian sources are *successively refinable* without loss, and therefore the two users can achieve, respectively, distortions $D_1 = D(R_1 + R_2)$ and $D_2 = D(R_2)$, i.e., the minimum distortion achievable at the total received rate. Note that for two users, a two-layer digital scheme is maximal, i.e. cannot be improved by adding layers.

Analog Communications

For a single-dimensional Gaussian source and channel, we find that the minimum distortion

$$D_{opt} = \lambda 2^{-2C} = \frac{\lambda}{1+SNR} \tag{5.6}$$

is achieved by a very simple uncoded (unit block length) system:

$$X = gS, \tag{5.7}$$

$$\hat{S} = \frac{g\lambda}{g^2\lambda + \sigma^2} Y. \tag{5.8}$$

The encoder applies a scaling factor $g = \sqrt{P/\lambda}$ to match available power, and the decoder performs linear least squares (LLS) estimation given the known noise variance. We assume that the statistic λ is communicated to the decoder via a side channel. Such encoder is often referred to in literature as Pulse Amplitude Modulation (PAM).

In this chapter, we consider a generalization of this analog system with the restriction that encoding and decoding are linear and allow for a side channel that communicates the subband variances as side information to the decoder. The encoder transmits $X = GS$ where G is an $M \times N$ matrix. Given a constraint on

average power (or per-component power by using the Hadamard) and the fact that we consider only the regime $M \leq N$, the optimal gain matrix G is diagonal (for $M < N$, the $N - M$ subbands with smallest variances are suppressed). Since the LLS decoder, optimal for linear decoders, depends on noise variance, the encoder can optimize the gain matrix expecting a specific channel SNR. Denoting the diagonal elements of G to be g_i and the normalized gain $d_i = g_i^2/P$, the distortion of the analog system can be written as

$$D(\text{SNR}) = \sum_{i=1}^{N} \frac{\lambda_i}{d_i \lambda_i \text{SNR} + 1} \tag{5.9}$$

subject to $\sum_i d_i \lambda_i = M$ and $d_i \geq 0$. This yields a water-filling result [58]:

$$D_{\text{ana}}(\text{SNR}) = \frac{(\sum_{i=1}^{K} \sqrt{\lambda_i})^2}{M\text{SNR} + K} + \sum_{i=K+1}^{N} \lambda_i, \tag{5.10}$$

where the number of subbands to be transmitted, K, is dependent on the channel SNR, much like in the digital case. The choice of K can be determined by solving a Lagrangian and increases monotonically with SNR up to M.

We briefly comment on the robustness of this analog scheme. Because the system is linear, a mismatch between design and actual SNR does not necessarily induce complete performance loss like in the digital case when coding near capacity. In fact, for reasonable mismatches, the performance may still be near optimal for the actual SNR. A rigorous analysis of robustness for the single-component case is given in [95].

5.3.2 Point-to-point Communication

We compare the end-to-end distortion performance between the proposed analog system and a digital system operating at capacity using the model in Section 5.3.1. We define the compression ratio to be $\beta = M/N$ and the performance ratio to be $\rho(\text{SNR}, \Lambda_S) = D_{\text{dig}}/D_{\text{ana}}$, where Λ_S is the covariance matrix of the source and D_{dig} and D_{ana} are the solutions to (5.3) and (5.10) for a target channel SNR. Note that in this setting, the digital system achieves optimal performance and therefore $\rho \leq 1$.

However, we are interested in how close does analog perform to this optimum.

A simple example is first presented to provide some intuition about how ρ may vary with SNR and Λ_S. A more general analysis of ρ for $\beta \leq 1$ is then discussed. We do not consider the case of $\beta > 1$ because it has been well-studied that linear analog encoding performs poorly compared to nonlinear coding such as space-filling [15] in this regime.

A Two-dimensional Source

Consider a system with $N = M = 2$, meaning that two source subbands are communicated with two channel uses per unit time. Assume the source covariance matrix is given by

$$\Lambda_S = \begin{pmatrix} 1 & 0 \\ 0 & \epsilon \end{pmatrix}.$$

For low SNR, the smaller source symbol is suppressed by both schemes, yielding distortions

$$D_{\text{dig}} = \frac{1}{(1+\text{SNR})^2} + \epsilon,$$

$$D_{\text{ana}} = \frac{1}{1+2\text{SNR}} + \epsilon.$$

In the limit of very low SNR, ρ approaches 1 regardless of ϵ, making the schemes comparable.

As SNR increases, the analog and digital systems transmit both source subbands when ϵ exceeds $1/(1+2\text{SNR})^2$ and $1/(1+\text{SNR})^2$ respectively. In this regime, the distortions become

$$D_{\text{dig}} = \frac{2\sqrt{\epsilon}}{1+\text{SNR}} \approx \frac{2\sqrt{\epsilon}}{\text{SNR}},$$

$$D_{\text{ana}} = \frac{(1+\sqrt{\epsilon})^2}{2\text{SNR}+2} \approx \frac{(1+\sqrt{\epsilon})^2}{2\text{SNR}}.$$

For SNR large, the performance ratio approaches the constant

$$\rho = \frac{4\sqrt{\epsilon}}{(1+\sqrt{\epsilon})^2}.$$

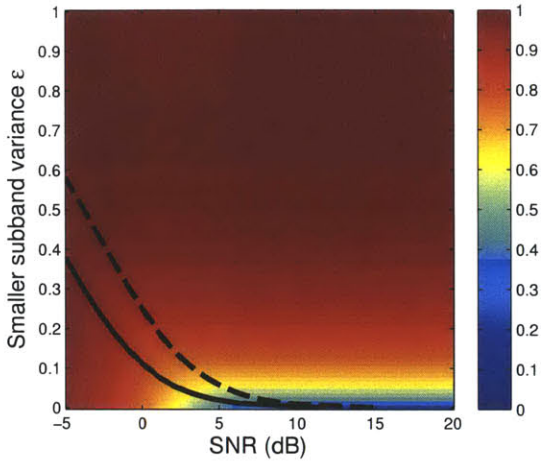


Figure 5-12: **Performance in point-to-point communication of a 2-dimensional source.** Plot of performance ratio ρ in terms of ϵ and SNR in point-to-point communication. The solid and dashed black lines correspond to the threshold for which two sources subbands are transmitted for the analog and digital cases respectively.

When $\epsilon = 1$, the ratio is also 1, which corresponds to the case of analog being optimal for the encoding of an iid Gaussian source. However, for general ϵ, the analog performance degrades monotonically with SNR and the loss can be very significant for small ϵ. Figure 5-12 demonstrates these trends.

Now consider the case when $M = 1$ meaning the channel has only one dimension. This means that the encoder must produce only one channel value for every $N = 2$ source values. In this case the analog coder must discard the weaker subband while the digital coder splits the available bit-rate between the two subbands. However, as ϵ decreases to 0, the effective source dimension becomes 1 and thus the analog system becomes optimal. This is illustrated in Figure 5-13. The green line on this plot corresponds to a cut at 10dB SNR of Figure 5-12, while the blue line shows the corresponding performance when $M = 1$. We observe that while analog is predictably optimal in the cases of $(\epsilon = 1, M = 2)$ and $(\epsilon = 0, M = 1)$, it is also competitive in a wide regime, in particular, admitting compression if the source is compressible.

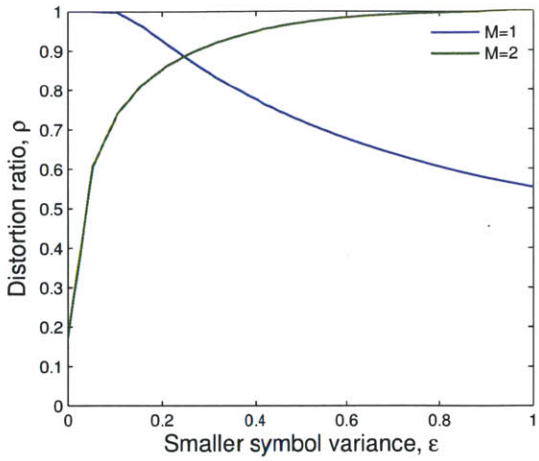


Figure 5-13: **Performance regimes in point-to-point communication.** Plot of performance ratio $\rho = \frac{D_{\text{dig}}}{D_{\text{ana}}}$ in terms of ϵ in point-to-point communication over a channel with 10dB SNR. The blue line corresponds to bandwidth compression at $M = 1$ while the green shows the case of bandwidth match at $M = 2$. When $\epsilon = 0$, the effective source dimension is 1 and thus analog is optimal.

A More General Case

We now consider the case when N and M are arbitrary but $M \leq N$ holds. Since the parameter space can be arbitrarily large, we restrict the source to a first-order Gauss-Markov (or AR(1)) process that is decorrelated using the Karhunen-Loève Transform (KLT) to form the vector S. We emphasize that the qualitative results hold more generally and this model is a tool to reduce the parameter space. Moreover, this model has strong ties to image and video compression due to the fact that the KLT and DCT are asymptotically equivalent [74].

We assume a source $S_{\text{GM}}(t) \in \mathbb{R}^N$ is independent in time but its components at a given instant form a stationary Gaussian process with a covariance matrix

$$(\Lambda_{\text{GM}})_{ij} = \alpha^{|i-j|}, \quad i, j = 1, 2, \ldots N.$$

The KLT of Λ_{GM} is a diagonal matrix Λ_S with entries corresponding to the eigenvalues of Λ_{GM}. They can be expressed as

$$\lambda_i = \frac{1 - \alpha^2}{1 - 2\alpha \cos\theta_i + \alpha^2}, \tag{5.11}$$

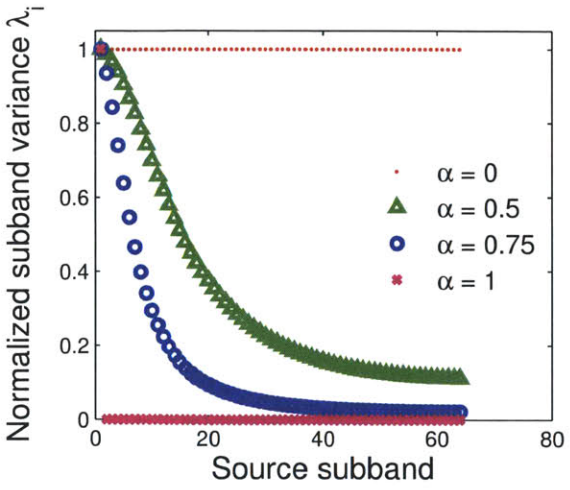


Figure 5-14: **Variance profile of the modelled source.** Plot of normalized subband variance λ_i for selected values of source correlation parameter α. The higher the correlation in the AR(1) process, the more compacted is the energy of the source signal in a small fraction of subbands.

where the θ_is are the positive roots of

$$\tan(N\theta) = -\frac{(1-\alpha^2)\sin\theta}{\cos\theta - 2\alpha + \alpha^2\cos\theta}. \tag{5.12}$$

These eigenvalues can be seen as taking non-uniform samples from the monotonically decreasing function

$$f(\theta;\alpha) = \frac{1-\alpha^2}{1-2\alpha\cos(\theta)+\alpha^2}$$

in the range $[0,\pi)$ [75]. Hence, the input to the encoder is S, a multi-variate memoryless Gaussian vector with decaying variances assumed in earlier sections. For reference, Fig. 5-14 shows how α affects the decay of λ.

When $M = N$, the performance results are well known. In the degenerate case of $\alpha = 0$, the system has no memory and this problem simplifies to that of an iid Gaussian source through AWGN channel with $\rho = 1$. For general α and high SNR, the ratio approaches the constant

$$\rho = N^2 \frac{(\prod_{i=1}^{N}\lambda_i)^{1/N}}{(\sum_{i=1}^{N}\sqrt{\lambda_i})^2},$$

which is the *Arithmetic Mean-Geometric Mean* performance gap discussed in [54].

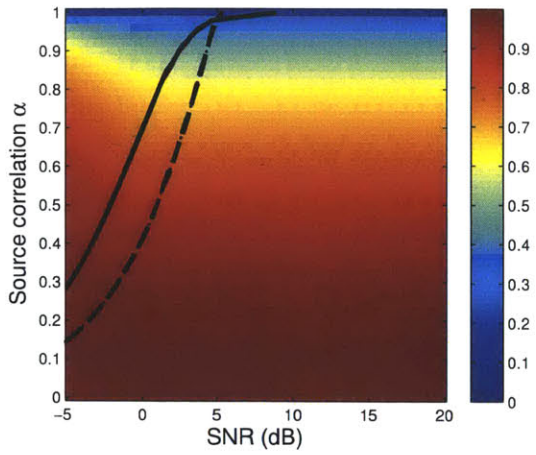
It is also well-known that $\rho \approx 1$ for very low SNR in general. As seen in the 2×2 case, both analog and digital schemes will only try to communicate the most significant components when the channel is very noisy. Since SNR and capacity are linearly related in this regime, analog communication performs as well as digital.

It may be tempting to say that for $\beta < 1$, these same trends hold. However, we show this is not the case and ρ does not necessarily behave in a monotonic fashion with respect to SNR. At one extreme, using the distortion results (5.3) and (5.10), we see that for high SNR, ρ decays as $(1 + \text{SNR})^{\beta}$ and hence the performance loss is dramatic. This is because a digital scheme can successfully compress N source symbols into M channel uses but an analog design must discard the $N - M$ components with smallest variances, which is inefficient. Meanwhile, for moderate SNR, we note that both analog and digital transmission may not transmit all source components possible. When the analog and digital transmit both communicate approximately M components, ρ can be close to 1. In this case, the analog scheme remains efficient because it is transmitting as many components as it can support with M channel uses while the digital is forced to suppress some subbands due to the noisiness of the channel, thereby reducing its asymptotic advantage.

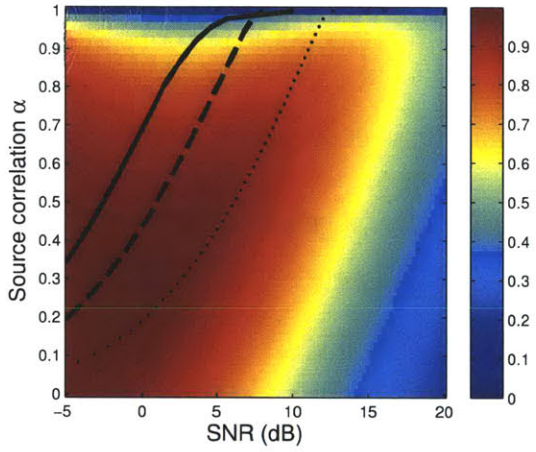
This is demonstrated in Figure 5-15, which looks at several choices of M for $N = 64$. We can note that for smaller β and moderate SNR, the analog design performs much better than asymptotic analysis may suggest. These results show that an analog design is reasonable beyond just the iid Gaussian case and the Arithmetic Mean-Geometric Mean performance gap does not hold except for $\beta = 1$. A designer may tolerate this amount of performance loss in favor of the simplicity and robustness of the analog system.

To better characterize the regime in which the analog system is competitive we fix the channel SNR at 10dB and show how the performance varies with α and β in Figure 5-16. In particular we can identify three regimes where the ratio ρ is close to 1:

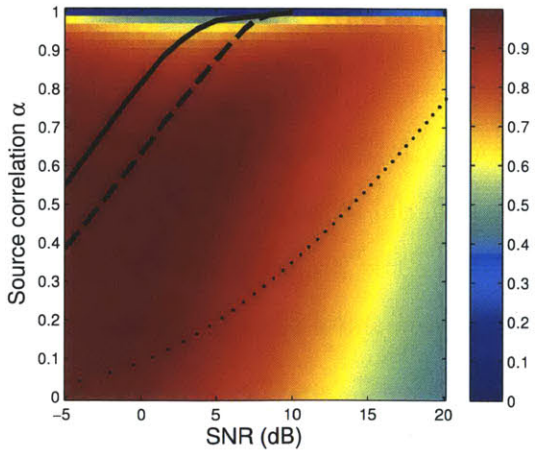
1. When $\beta = 1$ and $\alpha = 0$, the source is white and the system is matched, hence



(a) $M = 64$



(b) $M = 32$



(c) $M = 16$

Figure 5-15: **Performance regimes in point-to-point communication of a multi-dimensional source.** Plot of performance ratio ρ for $N = 64$ in terms of α and SNR in point-to-point communication. The solid line corresponds to the threshold for when the analog scheme transmits M subbands. The dashed and dotted lines correspond to the thresholds for when the digital scheme transmits M and N source subbands respectively.

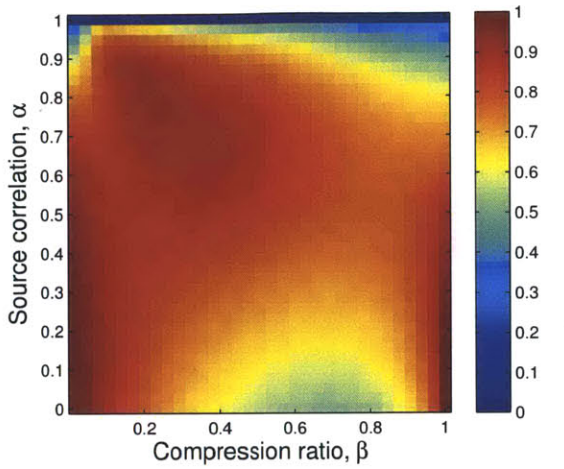


Figure 5-16: **Performance regimes in point-to-point communication of a multi-dimensional source.** Plot of performance ratio ρ in point-to-point communication over a channel with 10dB SNR in terms of α and β.

the analog is optimal. This effect spans a range where β is near 1 and α is low.

2. When β is very low, and α is small, the source is white and needs to be compressed into a very low bandwidth. This stresses both analog and digital systems and leads to similar performance.

3. When β is low and α is high, the source is correlated and needs to be compressed. In this regime, the linear system drops the weak subbands to fit the available bandwidth. This strategy turns out to be competitive with digital rate waterfilling.

The third regime is particularly interesting as it corresponds to most practical scenarios in which the source signal is highly correlated and the channel bandwidth is scarce. For instance, consider the broadcast of high-definition video over a wireless channel of $20MHz$. We inspect those scenarios more thoroughly in Section 5.4.

5.3.3 Broadcast Communication

We now consider the degraded broadcast case and compare the ideal purely-digital system (which combines successive refinement with superposition coding) and the analog linear system using the model in Section 5.3.1. The end-to-end distortion

performance is defined in terms of the distortions observed by each of the users and depends on channel SNR of each user. To comparatively evaluate the two systems, we use *total distortion*, i.e., sum of distortions of individual users. In general, one could use other metrics, e.g., the weighted sum. This overall distortion metric is a function of each user's channel SNR. For simplicity, we consider a scenario with only two users, and focus on three operational SNR regimes:

1. high SNR: $\text{SNR}_1 = 17\text{dB}$ and $\text{SNR}_2 = 10\text{dB}$
2. low SNR: $\text{SNR}_1 = 10\text{dB}$ and $\text{SNR}_2 = 3\text{dB}$
3. wide SNR: $\text{SNR}_1 = 17\text{dB}$ and $\text{SNR}_2 = 3\text{dB}$

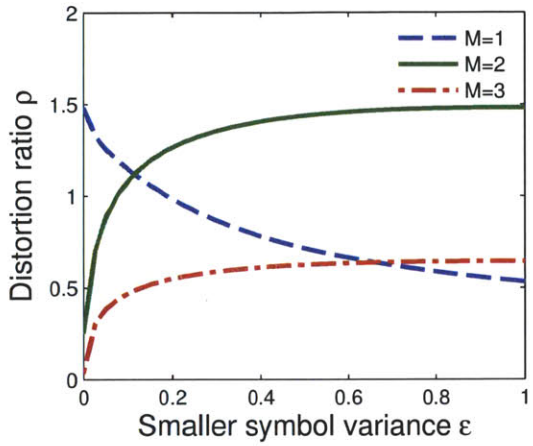
As in the case of point-to-point, one parameter in the evaluation is compression ratio, $\beta = M/N$. However, we also consider the case of bandwidth expansion, i.e., $\beta > 1$. Although the linear analog scheme cannot compete in this regime for point-to-point channels except for very low SNR, the broadcast setting gives it advantages over the digital system.

Thus the computed performance ratio is $\rho(\text{SNR}_{1,2}, \Lambda_S) = D_{\text{dig}}/D_{\text{ana}}$, where Λ_S is the covariance matrix of the source and D_{dig} and D_{ana} are the minimum total distortions achievable in either scheme when optimized over its tunable parameters according to Section 5.3.1. That is, in both cases the encoder decides how many source subbands to suppress and the digital superposition encoder optimizes the power distribution between the two bit-streams.

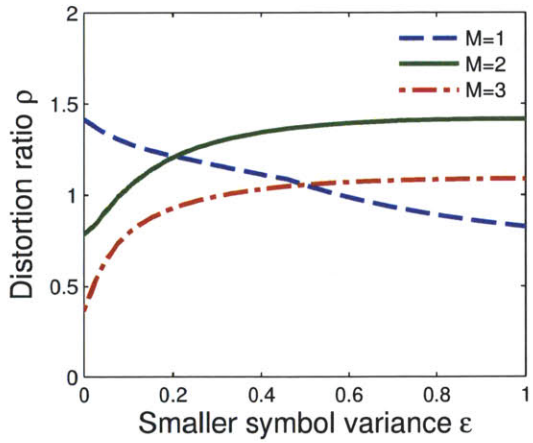
A Two-dimensional Source

Once again, we use a simple example to provide some intuition about how ρ may vary with the source covariance matrix. Consider the case of $N = 2$ and $M = 1, 2, 3$. As before, we assume $\lambda_1 = 1, \lambda_2 = \epsilon$.

Figure 5-17 plots the performance ratio ρ in terms of ϵ and the channel bandwidth M. We observe that how ρ changes with ϵ depends on the compression ratio. In the regime of bandwidth compression, when $M = 1$, the analog system delivers a



(a) $\mathrm{SNR}_1 = 17\mathrm{dB}, \mathrm{SNR}_2 = 10\mathrm{dB}$



(b) $\mathrm{SNR}_1 = 10\mathrm{dB}, \mathrm{SNR}_2 = 3\mathrm{dB}$

Figure 5-17: **Performance in broadcast communication of a 2-dimensional source.** Plot of the broadcast performance ratio ρ for the 2-dimensional source in terms of the smaller subband variance ϵ and channel bandwidth M. The three lines correspond to the case of bandwidth compression, match, and expansion.

performance gain for low ϵ, but a significant loss for white sources. In contrast, in the case of matched bandwidth $M = 2$, the gain of the analog system peaks at $\epsilon = 1$, as it converges to the optimal distortion bound. In the regime of bandwidth expansion, $M = 3$, the analog system, limited by its linearity, loses its broadcast advantage. The lower the ϵ, the worse performance it exhibits.

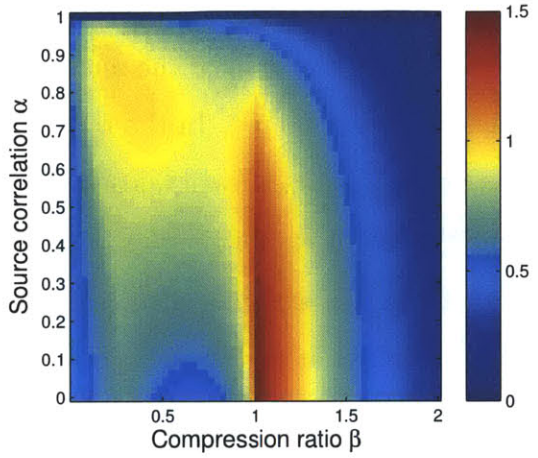
This points at the underlying determining factor of analog performance. When the source and channel are matched, it performs nearly optimally, but fails competitively to expand or compress the bandwidth. As ϵ tends to 0, the source can be approximated as single-dimensional by ignoring the low-variance subband. Thus, for $M = 1$ the system becomes matched. The performance gap (both loss and benefit) is more pronounced in the high SNR regime.

A More General Case

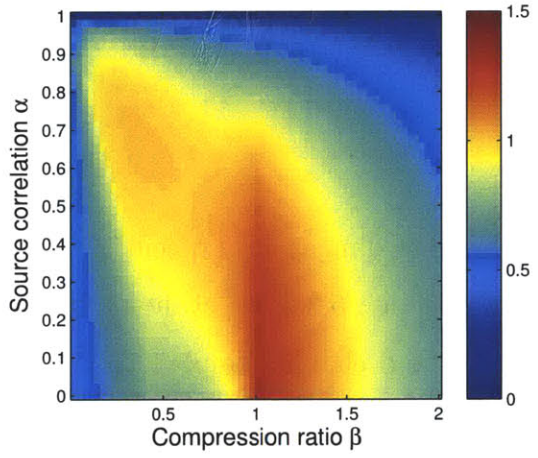
We now move to larger source dimensions, specifically $N = 64$. Using the same Gauss-Markov model as in Section 5.3.2, we evaluate the performance ratio ρ under varying source correlation α and channel bandwidth M. Once again, we look at a wide spectrum from bandwidth compression of 1/64, to bandwidth expansion of 2.

Figure 5-18 shows the performance ratio in the three considered SNR scenarios. We find that analog dominance is most pronounced in the area of matched bandwidth ($\beta = 1$) for low source correlation. Indeed, when the source is white ($\alpha = 0$), we expect the matched analog system to perform optimally, as discussed previously. However, we also find that analog remains dominant in the area of moderate compression and source correlation (top left in the plot). This aligns with our previous observations: if the source is correlated it can be "easily" compressed into the available channel bandwidth. The more correlated the source is, the lower the matching compression ratio.

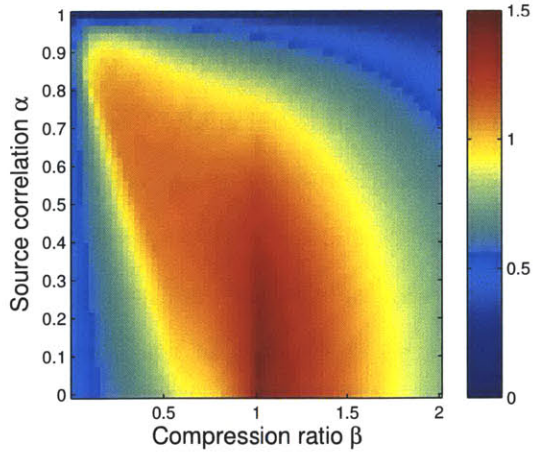
We find that, although the analog system shows the greatest maximum benefit in the high SNR regime, its performance gain decays abruptly with source-channel rate mismatch. On the other hand, if the SNR of the weak user is low, the analog system is superior in a larger domain. The modest difference between Fig. 5-18b and 5-18c



(a) SNR $= \{17, 10\}$dB



(b) SNR $= \{17, 3\}$dB



(c) SNR $= \{10, 3\}$dB

Figure 5-18: **Performance regimes in broadcast communication of a multi-dimensional source.** Broadcast performance ratio ρ for the three SNR regimes in terms of source correlation, α and compression ratio, β.

suggests that the dominating factor is the lower SNR rather than the SNR range. However, the overall benefit of the analog is generally diminished at higher SNR by the asymptotic gap. These results show that an analog design might offer substantial benefit in a broadcast scenario beyond the matched iid Gaussian case.

Observe that the layered digital design is not limited by the strict coder separation concept considered by Shannon in which the source and channel coders agree only on the rate of bits in the uniform lossless bit-pipe. Thus, although such design qualifies as JSCC, it is fundamentally limited by the bit-pipe. Gastpar et al. showed in [29] that if there is a match between the probability distribution of the source and the channel, for instance if rate-matched and white in our consideration above, then the simple analog, uncoded communication is optimal. However, as we have shown, even if they are not matched, the analog design can be superior to any digital method in particular operational regimes.

5.4 Discussion

Our results indicate a linear analog system might not necessarily perform as poorly as indicated by the asymptotic gap, even when compared to the optimal digital bound. Specifically, we found that when the available channel bandwidth is low and the encoder needs to compress, the analog system can retain its broadcast advantage if the source is correlated and thus compressible. In this section, we explore practical application scenarios to determine their operational regime. We also discuss additional benefits of the analog system when restricted by other constraints: computational complexity and latency.

5.4.1 Practical Applications

First, to benefit from analog communication, the source must have a degradable character. Examples of degradable signals include video, audio and sensors measurements such as environmental conditions (temperature, pressure, etc.).

Second, JSCC is naturally unsuitable if the channel coder is beyond our control,

hidden behind the layering abstraction of the communication system. For instance, the sheer diversity of the actual physical implementations of each link in a wide-area network renders JSCC infeasible. That said, the increasingly commonplace cross-layer or split design violates the layering and end-to-end principles on the wireless last hop. Such a design is motivated by the fact that physical layer characteristics of the wireless channel cannot be effectively concealed from the network. Examples include Snoop, split TCP and HTTP proxy at the wireless access point [5]. Thus, it is conceivable that an access point or base station would transcode a video stream from the format most suitable for IP networks to a format more suitable for wireless channel [31].

Furthermore, a JSCC approach does not necessarily discard the layering abstraction. A wireless PHY needs to deal with many practical issues specific to the device hardware, such as gain control, preamble detection, channel estimation, phase and amplitude equalization, carrier and sampling frequency offset compensation, or carrier sensing. Rather than recreating this custom functionality in the JSCC, the existing PHY could simply expose a more "raw" interface to the channel, as described in Chapter 4.

We note that this concept aligns with the efforts to expose more PHY-level information to the MAC layer in the form of per-subcarrier channel estimates or per-bit soft values [48,73,98]. To benefit from analog or hybrid transmission, the PHY needs to let the higher-layer (application-specific JSCC) perform constellation mapping.

Finally, all of our results are with respect to a particular distortion metric: MSE. Specific applications may have more relevant application-specific distortion metrics, yet MSE provide a general unifying approach to study all of these systems and the results are typically relevant to other metrics. Notably, PSNR the de facto metric for video content is MSE-based.

5.4.2 Operational Regime of Wireless Video Broadcast

Our results suggest that even purely linear schemes can be competitive in specific regimes. For example, one could apply the analog linear scheme described in Sec-

tion 5.3.1 to DCT frequencies (as source subbands) and treat the statistics as meta-data. For reference, the ATSC standard defines SDTV video resolution between 704×480 and 720×576 at 24 or 30fps, corresponding to $8e6$ and $12e6$ luminance values per second. An HDTV could bring the source bandwidth to $60e6$ with 2Mpixel screens.

In comparison, an 8MHz mobile TV (DVB-H) channel offers a net channel rate of around 10^6 dimensions per second. This is split across up to 15 TV stations, yielding 700×10^3 dim/s per source. For interlaced SDTV this indicates a compression ratio of 5:1 to 9:1. Similarly, 802.11a/g over the 20MHz channel sees net channel rate of 24×10^6 dim/s, but in practice utilizes only 50%-70% depending on the packet length. Streaming HDTV content over such channel would fall in the compression regime. Practical operational SNR range is $[-5, 30]$dB for DVB-H and $[3, 25]$dB for 802.11a/g. Thus, the parameters of practical scenarios of wireless transmission of video lie within the compression region where the linear analog scheme is competitive with the optimal digital bound in point-to-point and can be superior in the broadcast case. That is assuming significant correlation in the source, which is characteristic to video content.

5.4.3 Other Benefits of Analog Coding

In this chapter, we have compared the linear analog scheme to the optimal digital approach imposing only a bandwidth and signal power requirement. However, practical systems are limited in computational complexity and processing latency which restricts the blocklength and reduces performance. Thus, the simplicity and very low latency of the linear analog system could make it the preferred approach in power-restricted applications, such as mobile or sensor networks. For instance, discarding space-time coding, the convolutional FEC consumes nearly half of the digital circuit power in 802.11n [42]. Similarly, in the source coder, an analog scheme might not need to perform expensive motion search but can resort to one-pass vector arithmetic.

Chapter 6

Related Work

Recent years have witnessed much interest in making video quality scale with channel quality [10,64,67,102]. The general approach so far has been to divide the video stream into a base layer that is necessary for decoding the video, and an enhancement layer that improves its quality [13,19,32,35,39,53,85]. Proposals in this area differ mainly in how they generate the two layers and the code they use to protect them. For example, some proposals consider the I frames as the base layer and the P and B frames as the enhancement layer [107]. More recent approaches create a base layer by quantizing the video to a coarse representation, which is refined by the enhancement layers [32,85]. Given video layers of different importance, one has many choices for protecting them unequally. Some proposals put more FEC coding on the base layer than the enhancement layers [19,39]. Others employ embedded diversity coding, where a high-rate code allows the enhancement layer to harness good channel realizations, while the embedded high-diversity code provides guarantees that at least the base layer is received reliably [2,20,32]. Hierarchical modulation and superposition coding are examples of this approach [16,56]. Motivated by this prior work, SoftCast takes scalable video one step further; it disposes of the coarse granularity of layers in favor of a continuously scalable design.

Related work also includes analog and digital TV. Analog television also linearly transforms the luminance values for transmission. And, in fact, analog television also shares the property that the quality of the transmitted video degrades smoothly as

the channel quality degrades. A key advantage of our approach is that our encoding scheme also leverages the powerful digital computation capabilities (which became available subsequent to the development of analog television) to encode the video both for compression and error protection. Hence, we can obtain transmission efficiency comparable to standard digital video coding schemes such as MPEG.

Digital TV also deals with video multicast [76]. The focus in digital TV however is on ensuring a minimum video quality to all receivers rather than on ensuring that each receiver obtains the best video quality supported by its channel. Further, the variability in channel quality is lower because there is neither mobility nor interference. In fact, proposals for extending Digital TV to mobile handheld devices argue for graceful degradation and propose to employs a 2-layer video with hierarchical modulation [56].

There is a large body of work that allows a source to adapt its transmission bitrate to a mobile receiver [11,37,51,98]. However, these schemes require fast feedback, and are limited to a single receiver. Further, they need to be augmented with additional mechanisms to adapt the video codec rate to fit within the available bitrate. In contrast, SoftCast provides a unified design that eliminates the need to adapt bitrate and video coding at the source, and instead allows the receiver to extract a video quality that matches its instantaneous channel.

This work builds on past work in information theory on rate distortion and joint source and channel coding (JSCC) [16]. This past work however mainly focuses on theoretical bounds. The fact that white Gaussian sources transmitted uncoded over white Gaussian channels can be optimal has been known for decades [7,34]. Other combinations of sources and channels where uncoded transmission is optimal are also explored [29]. The Gaussian match was generalized to linear encoding in the vector case [58] and nonlinear encoding using space-filling curves when the number of source components does not match the number of channel uses per unit time [15,38]. The problem of bandwidth expansion and compression has also been addressed by hybrid systems combining digital techniques with linear analog encoding [77,88], although the scope of analysis was limited to a 2-dimensional white source. Correlation in the

source was shown to aid the broadcast scenario in a hybrid scheme proposed in [72].

Of particular interest are investigations of analog or hybrid systems that transmit media content over AWGN channels. In [54], wireless image transmission is made more robust using an analog transmitter. In [55], a digital base layer is refined with an analog enhancement layer. In [18], a nonlinear analog image encoder is designed to achieve graceful degradation of quality with channel SNR. Unlike those schemes, the design of SoftCast and RawOFDM covers all aspects of wireless transmission of video thus allowing experimental evaluation in a physical testbed.

Finally, SoftCast leverages a rich literature in signal and image processing, including decorrelation transforms such as 3D DCT [14,71], the least square estimator [57], the Hadamard transform [6], and optimal linear transforms [58]. SoftCast uses these tools in a novel PHY-video architecture to deliver a video quality that scales smoothly with channel quality.

Chapter 7

Conclusion

Although Shannon's separation principle permeates the layered design of computer communication systems, it is fundamentally limiting its performance in increasingly common scenarios which invalidate its basic assumptions. In particular, the digital mapping of source signal to bits and then back to a waveform signal on the channel is inherently suboptimal when streaming degradable content such as video over a mobile and broadcast wireless channel.

I designed SoftCast, a clean-slate design for wireless video. SoftCast enables a video source to broadcast a single stream that each receiver decodes into a video quality commensurate with its instantaneous channel quality. Further, SoftCast requires no receiver feedback, bit rate adaptation, or video code rate adaptation. SoftCast adopts an integrated design for video and PHY layer coding, making the whole network stack act as a linear transform. I showed that such a design improves the video quality for multicast users, eliminates video glitches caused by mobility, and increases robustness to interference and channel errors.

To integrate SoftCast in the network stack, I proposed RawPHY, a new architecture for the wireless PHY layer that exposes a waveform interface to the channel while canceling most of non-additive effects of the channel. RawPHY not only allows easy integration of SoftCast alongside traditional data applications, but also enables flexible error protection by shifting the control over the throughput vs. robustness trade-off from PHY to the application.

To provide insight into the performance of SoftCast, I demonstrated through analysis that an analog-like scheme with linear encoding and decoding can be an efficient way to communicate multivariate Gaussian sources in terms of distortion, latency, robustness and computational complexity. Specifically, in the broadcast setting in regimes of limited channel bandwidth and highly compressible source signals, the analog scheme outperforms the digital design regardless of the computational effort employed by the digital system.

In its current design, SoftCast has limitations. Specifically, SoftCast is unsuitable for sources that can be efficiently fitted by nonlinear models. Consider for example a white ball moving on a black background. A video codec that uses motion compensation, such as MPEG, can encode each frame by simply sending the shift of the ball's center, which is more efficient than using a linear transform like 3D DCT to compress the video. Such videos however are atypical and if they arise they can be transmitted using standard video coding. For videos of natural scenes, linear transforms like DCT and Wavelets are highly effective in compressing the video information [71,104]. Furthermore, the gains of SoftCast arise mainly from its robustness to channel errors and packet loss. In contrast, existing nonlinear video codecs are highly sensitive to errors. I believe that a better tradeoff can be reached if we can leverage the intrinsic resilience of video signals to deal with errors on the channel. In general, the tradeoffs between the gains from efficient but error-sensitive compression and the cost of error correction codes and packet retransmission at the lower layers are important research topics for the future of mobile and broadcast video.

Appendix A

Proofs

A.1 Proof of Lemma 3.3.1

We want to determine the set of optimal linear scaling factors for each chunk that minimizes the expected reconstruction error (computed as mean square error), while within the total power constraint, as inspired by [58]. Note that, since the DCT transform is orthogonal, the reconstruction error of chunk i ($x_i[1 \ldots N]$, where N is the number of DCT components in a chunk) is directly proportional to the reconstruction error in the video frame.

Let us model the channel as one with additive white noise. Thus, for each value in chunk i, $x_i[j]$, we transmit $y_i[j] = g_i x_i[j]$, and the receiver receives $\hat{y}_i[j] = y_i[j] + n$, where g_i is the linear scaling factor for this chunk, and n is a random variable with zero-mean and specific variance, σ (the same for all chunks). Subsequently, the receiver decodes

$$\hat{x}_i[j] = \frac{\hat{y}_i[j]}{g_i} = x_i[j] + \frac{n}{g_i}.$$

The expected mean square error is:

$$err = E\left[\sum_i (\sum_j (\hat{x}_i[j] - x_i[j])^2)\right] = N \sum_i \frac{E[n^2]}{g_i^2} = N \sum_i \frac{\sigma^2}{g_i^2}$$

Clearly, the best scaling factor would be infinite, if not for the power constraint. Let

$\lambda_i = E[x_i^2]$ be the power of chunk i ($x_i[1 \ldots N]$), $\mu_i = E[y_i^2]$ be its power after applying the scaling factor, and P the total power budget. We can drop N in the minimand since it is merely a constant factor, and formally rewrite the problem as follows.

$$\begin{aligned} \min err &= \sigma^2 \sum_i \frac{\lambda_i}{\mu_i} \\ \text{subject to:} \quad & \sum_i \mu_i \leq P \text{ and } \mu_i \geq 0 \end{aligned} \tag{A.1}$$

We can solve this optimization using the technique of Lagrange multipliers. The Lagrangian is

$$L = \sigma^2 \sum_i \frac{\lambda_i}{\mu_i} + \gamma \left(\sum_i \mu_i - P \right)$$

Differentiating separately by μ_i and γ and setting to 0, yields:

$$\begin{aligned} \sqrt{\gamma} &= \sum_i \sqrt{\lambda_i \sigma^2}/P \\ \mu_i &= \sqrt{\frac{\lambda_i \sigma^2}{\gamma}} = P \frac{\sqrt{\lambda_i}}{\sum_i \sqrt{\lambda_i}} \\ g_i &= \sqrt{\frac{\mu_i}{\lambda_i}} = \sqrt{\frac{P}{\sqrt{\lambda_i} \sum_i \sqrt{\lambda_i}}} \end{aligned}$$

The optimal scaling factor for each chunk is therefore such that the resulting power of the row is proportional to the *square root* of its original power. Some readers might find it more intuitive that the optimal solution should completely equalize the resulting power, i.e. $\mu_i = P/k$; but substituting this in (A.1) shows otherwise.

Bibliography

[1] Cisco visual networking index: Forecast and methodology 2008-2013. `http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html`, June 2009.

[2] Ahmed B. Abdurrhman, Michael E. Woodward, and Vasileios Theodorakopoulos. *Robust Transmission of H.264/AVC Video Using 64-QAM and Unequal Error Protection*, pages 117–125. 2009.

[3] T. Ahmed, G. Buridant, and A. Mehaoua. Encapsulation and marking of mpeg-4 video over ip differentiated services. In *Computers and Communications, 2001. Proceedings. Sixth IEEE Symposium on*, pages 346–352. IEEE, 2001.

[4] Guangwei Bai and Carey Williamson. The Effects of Mobility on Wireless Media Streaming Performance. In *Proceedings of Wireless Networks and Emerging Technologies*, pages 596–601, july 2004.

[5] Hari Balakrishnan, Venkata N. Padmanabhan, Srinivasan Seshan, and Randy H. Katz. A comparison of mechanisms for improving TCP performance over wireless links. In *IEEE/ACM Trans. on Networking, 5(6)*, pages 756–769, 1997.

[6] K. Beauchamp. *Applications of Walsh and Related Functions.* Academic Press, 1984.

[7] T. Berger. *Rate Distortion Theory.* Prentice-Hall, Englewood Cliffs, NJ, 1971.

[8] C. Berrou, Glavieux A., and Thitimajshima P. Near Shannon limit error-correcting coding and decoding: turbo-codes. In *Proc. ICC*, volume 93, pages 1064–1070, 1993.

[9] V.G. Bose. *Design and implementation of software radios using a general purpose processor.* PhD thesis, MIT, 1999.

[10] J.W. Byers, M. Luby, and M. Mitzenmacher. A digital fountain approach to asynchronous reliable multicast. *IEEE JSAC*, 20(8), Oct 2002.

[11] Joseph Camp and Edward W. Knightly. Modulation rate adaptation in urban and vehicular environments: cross-layer implementation and experimental evaluation. In *Proc. of ACM MOBICOM*, 2008.

[12] M. Castro, P. Druschel, A.M. Kermarrec, and A.I.T. Rowstron. Scribe: A large-scale and decentralized application-level multicast infrastructure. *Selected Areas in Communications, IEEE Journal on*, 20(8):1489–1499, 2002.

[13] G. Cheung and A. Zakhor. Bit allocation for joint source/channel coding of scalable video. *IEEE Trans. Image Processing*, 9(3), March 2000.

[14] C.H. Chou and C.W. Chen. A perceptually optimized 3-d subband codec for video communication over wireless channels. *Circuits and Systems for Video Technology, IEEE Transactions on*, 6(2):143–156, 1996.

[15] S.Y. Chung. *On the construction of some capacity-approaching coding schemes.* PhD thesis, MIT, 2000.

[16] T. Cover and J. Thomas. *Elements of Information Theory.* 1991.

[17] T. M. Cover. Broadcast channels. *IEEE Trans. Inform. Theory*, 18(1):2–14, January 1972.

[18] Helge Coward. *Joint source-channel coding: Development of methods and utilization in image communications.* PhD thesis, Norwegian University of Science and Technology, 2002.

[19] D. Dardari, M. G. Martini, M. Mazzotti, and M. Chiani. Layered video transmission on adaptive ofdm wireless systems. *EURASIP J. Appl. Signal Process.*, 2004:1557–1567, 2004.

[20] S.N. Diggavi, A.R. Calderbank, S. Dusad, and N. Al-Dhahir. Diversity embedded space–time codes. *IEEE Transactions on Information Theory*, 54, Jan 2008.

[21] W.H.R. Equitz and T.M. Cover. Successive refinement of information. *Information Theory, IEEE Transactions on*, 37(2):269–275, 1991.

[22] U. Erez, M.D. Trott, and G.W. Wornell. Rateless coding for Gaussian channels. *Arxiv preprint arXiv:0708.2575*, 2007.

[23] ETSI. *Digital Video Broadcasting; Framing structure, channel coding and modulation for digital terrestial television*, Jan 2009. EN 300 744.

[24] FFmpeg. `http://www.ffmpeg.com`.

[25] Fftw home page. `http://www.fftw.org/`.

[26] S. Floyd, V. Jacobson, C.G. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking (TON)*, 5(6):784–803, 1997.

[27] Didier Le Gall. MPEG: a video compression standard for multimedia applications. *Commun. ACM*, 34(4):46–58, 1991.

[28] R. Gallager. Low-density parity-check codes. *Information Theory, IRE Transactions on*, 8(1):21–28, 1962.

[29] M. Gastpar, B. Rimoldi, and M. Vetterli. To code, or not to code: Lossy source–channel communication revisited. *IEEE Trans. Inform. Theory*, 49(5):1147–1158, 2003.

[30] M. Gastpar, M. Vetterli, and P. L. Dragotti. Sensing reality and communicating bits: A dangerous liaison. *IEEE Sig. Process. Mag.*, 23(4):70–83, July 2006.

[31] Noam Geri. Wireless uncompressed HDTV solves problems that compression cannot. *EE Times*, May 2007.

[32] M. M. Ghandi, B. Barmada, E. V. Jones, and M. Ghanbari. H.264 layered coded video over wireless networks: Channel coding and modulation constraints. *EURASIP J. Appl. Signal Process.*, 2006.

[33] The GNU software radio. `http://gnuradio.org/trac`.

[34] T. J. Goblick, Jr. Theoretical limitations on the transmission of data from analog sources. *IEEE Trans. Inform. Theory*, IT-11(4):558–567, October 1965.

[35] R. Hamzaoui, V. Stankovic, and Z. Xiong. Optimized error protection of scalable image bit streams [advances in joint source-channel coding for images]. *Signal Processing Magazine, IEEE*, 22(6):91–107, 2005.

[36] J. Heiskala and J. Terry. *OFDM Wireless LANs: A Theoretical and Practical Guide.* Sams Publishing, 2001.

[37] G. Holland, N. Vaidya, and P. Bahl. A rate-adaptive MAC protocol for multi-hop wireless networks. In *Proc. of ACM MOBICOM*, pages 236–251, 2001.

[38] Y. Hu, J. Garcia-Frias, and M. Lamarca. Analog Joint Source Channel Coding Using Space-Filling Curves and MMSE Decoding. In *Data Compression Conference*, pages 103–112. IEEE Computer Society, 2009.

[39] Chung-Lin Huang and Sling Liang. Unequal error protection for MPEG-2 video transmission over wireless channels. In *Signal Process. Image Commun.*, 2004.

[40] D.A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[41] ITU-T. *Advanced video coding for generic audiovisual services*, May 2003. ITU-T Recommendation H.264.

[42] Balaji V. Iyer and Thomas M. Conte. On power and energy trends of IEEE 802.11n PHY. In *ACM MSWiM*, 2009.

[43] Szymon Jakubczak and Dina Katabi. Softcast: clean-slate scalable wireless video. In *Proceedings of the 2010 ACM workshop on Wireless of the students, by the students, for the students*, S3 '10, pages 9–12, New York, NY, USA, 2010. ACM.

[44] Szymon Jakubczak and Dina Katabi. Softcast: One-size-fits-all wireless video. In *SIGCOMM Demo Session*, 2010.

[45] Szymon Jakubczak and Dina Katabi. Softcast: Clean-slate scalable wireless video. Technical Report MIT-CSAIL-TR-2011-008, CSAIL, MIT, Mar 2011.

[46] Szymon Jakubczak, Hariharan Rahul, and Dina Katabi. One-size-fits-all wireless video. In *HotNets*, 2009.

[47] Szymon Jakubczak, John Z. Sun, Dina Katabi, and Vivek Goyal. Performance regimes of uncoded linear communications over AWGN channels. In *45th Conf. on Information Sciences and Systems (CISS)*, Mar 2011.

[48] Kyle Jamieson and Hari Balakrishnan. PPR: Partial packet recovery for wireless networks. In *Proc. of ACM SIGCOMM 2007, Kyoto, Japan.*

[49] Hong Jiang and P.A. Wilford. A hierarchical modulation for upgrading digital broadcast systems. 51, June 2005.

[50] SVC reference software. `http://ip.hhi.de/imagecom_G1/savce/downloads/SVC-Reference-Software.htm`.

[51] Glenn Judd, Xiaohui Wang, and Peter Steenkiste. Efficient channel-aware rate adaptation in dynamic environments. In *Proc. of ACM MobiSys*, 2008.

[52] S. Katti, D. Katabi, H. Balakrishnan, and M. Medard. Symbol-level network coding for wireless mesh networks. *ACM SIGCOMM Computer Communication Review*, 38(4):401–412, 2008.

[53] L.P. Kondi, F. Ishtiaq, and A.K. Katsaggelos. Joint source-channel coding for motion-compensated dct-based snr scalable video. *Image Processing, IEEE Transactions on*, 11(9):1043–1052, 2002.

[54] I. Kozintsev and K. Ramchandran. A wavelet zerotree-based hybrid compressed/uncompressed framework for wireless image transmission. In *31st Asilomar Conf. Sig., Sys., & Computers*, November 1997.

[55] Igor Kozintsev and Kannan Ramchandran. Robust image transmission over energy-constrained time-varying channels using multiresolution joint source-channel coding. *IEEE Trans. Signal Processing*, 46(4), April 1998.

[56] T. Kratochvíl. *Multi-Carrier Systems & Solutions*, chapter Hierarchical Modulation in DVB-T/H Mobile TV Transmission, pages 333–341. 2009.

[57] Charles Lawson and Richard Hanson. *Solving Least Squares Problems.* Society for Industrial Mathematics, 1987.

[58] Kyong-Hwa Lee and Daniel P. Petersen. Optimal linear coding for vector channels. *IEEE Trans. Communications*, Dec 1976.

[59] Y. Linde, A. Buzo, and R. Gray. An algorithm for vector quantizer design. *Communications, IEEE Transactions on*, 28(1):84–95, 1980.

[60] S.P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, IT-28:129–137, Mar 1982.

[61] M. Loève. *Probability theory.* Number 46 in Graduate Texts in Mathematics. Springer-Verlag, 1978.

[62] David J.C. MacKay. *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press, 2003.

[63] D.J.C. MacKay and R.M. Neal. Near Shannon limit performance of low density parity check codes. *Electronics letters*, 33(6):457–458, 1997.

[64] A. Majumdar, D.G. Sachs, I.V. Kozintsev, K. Ramchandran, and M.M. Yeung. Multicast and unicast real-time video streaming over wireless lans. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(6):524–534, 2002.

[65] Miguel O. Martínez-Rach et al. Quality assessment metrics vs. PSNR under packet loss scenarios in MANET wireless networks. In *MV: Intl. workshop on mobile video*, 2007.

[66] J. Max. Quantizing for minimum distortion. *IEEE Transactions on Information Theory*, IT-6:7–12, Mar 1960.

[67] S. McCanne, M. Vetterli, and V. Jacobson. Low-complexity video coding for receiver-driven layered multicast. *IEEE JSAC*, 15(6):983–1001, Aug 1997.

[68] U. Mittal and N. Phamdo. Hybrid digital-analog (HDA) joint source-channel codes for broadcasting and robust communications. *IEEE Trans. Information Theory*, 48(5):1082–1102, May 2002.

[69] Limelight Networks. Financial analyst meeting. `http://www.limelightnetworks.com/pdfs/2010/2010FinancialAnalystDay.pdf`, 2010.

[70] W. Perera. Architectures for multiplierless fast Fourier transform hardware implementation in VLSI. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 35(12):1750–1760, Dec 1987.

[71] C.I. Podilchuk, N.S. Jayant, and N. Farvardin. Three-dimensional subband coding of video. 4(2), Feb 1995.

[72] Vinod Prabhakaran, Rohit Puri, and Kannan Ramchandran. Hybrid digital-analog strategies for source-channel broadcast. In *Allerton Conference*, 2005.

[73] Hariharan Rahul, Farinaz Edalat, Dina Katabi, and Charles Sodini. Frequency-aware rate adaptation and MAC protocols. In *Proc. of ACM MOBICOM*, Sep 2009.

[74] K. R. Rao and P. Yip. *Discrete Cosine Transform: Algorithms, Advantages, Applications.* Academic Press, San Diego, CA, 1990.

[75] W. D. Ray and R. M. Driver. Further decomposition of the Karhunen-Loève series representation of a stationary random process. *IEEE Trans. Inform. Theory*, IT-16(6):663–668, November 1970.

[76] UH Reimers. DVB - the family of international standards for digital video broadcasting. *Proc. of the IEEE*, 94(1):173–182, 2006.

[77] Z. Reznic, M. Feder, and R. Zamir. Distortion bounds for broadcasting with bandwidth expansion. *IEEE Trans. Information Theory*, 52(8):3778–3788, Aug. 2006.

[78] I.E.G. Richardson. *H. 264 and MPEG-4 video compression: video coding for next-gen multimedia.* John Wiley & Sons Inc, 2003.

[79] David Salomon. *Guide to Data Compression Methods.* Springer, 2002.

[80] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Trans. Comput. Syst.*, 2:277–288, November 1984.

[81] Sandvine. Global Internet phenomena report. `http://www.wired.com/images_blogs/epicenter/2011/05/SandvineGlobalInternetSpringReport2011.pdf`, 2011.

[82] A.D. Sarwate. Robust and adaptive communication under uncertain interference. Technical Report UCB/EECS-2008-86, University of California at Berkeley, 2008.

[83] T.M. Schmidl and D.C. Cox. Robust frequency and timing synchronization for ofdm. *Communications, IEEE Transactions on*, 45(12):1613–1621, 1997.

[84] Martin Schnell. Hadamard codewords as orthogonal spreading sequences in synchronous DS CDMA systems for mobile radio channels. In *IEEE Intl. Symposium on Spread Spectrum Techniques and Applications*, 1994.

[85] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H.264/AVC standard. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2007.

[86] C. E. Shannon. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27:379–423, July 1948. Continued 27:623–656, October 1948.

[87] C. E. Shannon. Two-way communication channels. In *the 4th Berkeley Symp. Math. Satist. Probability*, 1961.

[88] M. Skoglund, N. Phamdo, and F. Alajaji. Hybrid digital-analog source-channel coding for bandwidth compression/expansion. *IEEE Trans. Information Theory*, 52(8):3757–3763, 2006.

[89] M. Slanina, T. Kratochvíl, and V. Říčny. Robustness of compressed high definition video to transmission packet loss. In *ELMAR, 2010 PROCEEDINGS*, Sep 2010.

[90] Spiral project: Viterbi decoder software generator. `http://www.spiral.net`.

[91] David Tse and Pramod Viswanath. *Fundamentals of Wireless Communication.* Cambridge University Press, 2005.

[92] E. Tuncel and K. Rose. Additive successive refinement. *Information Theory, IEEE Transactions on*, 49(8):1983–1991, 2003.

[93] G. Ungerboeck. Channel coding with multilevel/phase signals. *Information Theory, IEEE Transactions on*, 28(1):55–67, 1982.

[94] Universal software radio peripheral 2. `http://www.ettus.com/downloads/ettus_ds_usrp2_v2.pdf`.

[95] L. R. Varshney. Optimal information storage: Nonsequential sources and neural channels. Master's thesis, Mass. Inst. Tech., June 2006.

[96] S. Vembu, S. Verdu, and Y. Steinberg. The source-channel separation theorem revisited. In *IEEE Trans. Inf. Theory*, 1995.

[97] Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IT–13:260–269, April 1967.

[98] Mythili Vutukuru, Hari Balakrishnan, and Kyle Jamieson. Cross-layer wireless bit rate adaptation. In *Proc. of ACM SIGCOMM*, Aug 2009.

[99] A. Watson. Image compression using the discrete cosine transform. *Mathematica Journal*, 4:81–88, Jan. 1994.

[100] M. Wien, H. Schwarz, and T. Oelbaum. Performance analysis of SVC. *IEEE Trans. Circuits and Systems for Video Technology*, 17(9), Sept. 2007.

[101] Grace Woo, Pouya Kheradpour, and Dina Katabi. Beyond the bits: Cooperative packet recovery using phy information. In *Proc. of ACM MobiCom 2007, Montreal, Canada.*

[102] Dapeng Wu, Y.T. Hou, and Ya-Qing Zhang. Scalable video coding and transport over broadband wireless networks. *Proc. of the IEEE*, 89(1), 2001.

[103] x264 - a free H.264/AVC encoder. `http://www.videolan.org/developers/x264.html`.

[104] Zixiang Xiong, K. Ramchandran, M.T. Orchard, and Ya-Qin Zhang. A comparative study of dct- and wavelet-based image coding. *IEEE Trans. on Circuits and Systems for Video Technology*, Aug 1999.

[105] Xiph.org media. `http://media.xiph.org/video/derf/`.

[106] J. Xu, Z. Xiong, S. Li, and Y.-Q. Zhang. Memory-constrained 3-D wavelet transform for video coding without boundary effects. *IEEE Trans. on Circuits and Systems for Video Technology*, 12, 2002.

[107] Xiaofeng Xu, M. van der Schaar, S. Krishnamachari, Sunghyun Choi, and Yao Wang. Fine-granular-scalability video streaming over wireless LANs using cross layer error control. In *IEEE ICASSP*, May 2004.

[108] S. Yoo, S. Yoon, S.Y. Kim, and I. Song. A novel PAPR reduction scheme for OFDM systems: Selective mapping of partial tones (SMOPT). *Consumer Electronics, IEEE Transactions on*, 52(1):40–43, 2006.